Optimisation of Production From an Oil-Reservoir Using Augmented Lagrangian Methods.

PhD Thesis

Daniel Christopher Doublet

Department of Mathematics
University of Bergen

November 2007

Abstract

This work studies the use of augmented Lagrangian methods for water flooding production optimisation from an oil reservoir. Commonly, water flooding is used as a means to enhance oil recovery, and due to heterogeneous rock properties, water will flow with different velocities throughout the reservoir. Due to this, water breakthrough can occur when great regions of the reservoir are still unflooded so that much of the oil may become ”trapped” in the reservoir. To avoid or reduce this problem, one can control the production so that the oil recovery rate is maximised, or alternatively the net present value (NPV) of the reservoir is maximised.

We have considered water flooding, using smart wells. Smart wells with down-hole valves gives us the possibility to control the injection/production at each of the valve openings along the well, so that it is possible to control the flow-regime. One can control the injection/production at all valve openings, and the setting of the valves may be changed during the production period, which gives us a great deal of control over the production and we want to control the injection/production so that the profit obtained from the reservoir is maximised.

The problem is regarded as an optimal control problem, and it is formulated as an augmented Lagrangian saddle point problem. We develop a method for optimal control based on solving the Karush-Kuhn-Tucker conditions for the augmented Lagrangian functional, a method, which to my knowledge has not been presented in the literature before.

The advantage of this method is that we do not need to solve the forward problem for each new estimate of the control variables, which reduces the computational effort compared to other methods that requires the solution of the forward problem every time we find a new estimate of the control variables, such as the adjoint method.

We test this method on several examples, where it is compared to the adjoint method. Our numerical experiments show that the method is convergent, and when comparing it to the adjoint method it converges faster, in terms of computational effort, than the adjoint method.

Furthermore, the method is also applied to a history matching problem, where we estimate permeabilities. Also for this problem, the method shows good results.

Moreover, we consider the augmented Lagrangian method, for optimal control problems. We study several marching schemes, that have previously been used to solve optimal control problem, and we develop a new marching scheme. Examples are tested with the new marching scheme and compared to other known marching schemes. From our experiments, we see that the scheme is convergent and that it find higher NPV than the marching schemes which we compare it to.

Preface

This dissertation is financed by the Norwegian Research Council, Petromaks programme, project No. 163383/S30, and the aim is to investigate the possibility of using augmented Lagrangian methods to solve optimal control problems which occur in connection with oil production, more concrete optimisation of water flooding of reservoirs and history matching. The work was started in November 2004, and the main adviser is Sigurd I. Aanonsen, while Xue-Cheng Tai and Trond Mannseth are co-advisers. During this three-year period, the work has been conducted at the Centre for Integrated Petroleum Research (CIPR) and at the Department of Mathematics at the University of Bergen. The main new contribution is the development, testing and application of a method for optimal control based on solving the Karush-Kuhn-Tucker conditions for the augmented Lagrangian functional, a method, which to my knowledge has not been presented in the literature before.

The thesis consists of two parts, where the first part explains some theory related to the problems we are studying and the mathematical algorithms we are using. The second part of the thesis consists of papers that have been written during my time at CIPR, which relates to part one of the thesis. A brief outline of the thesis, is as follows,

Part I:

Chapter 1 explains the motivation and the main objective of the study done in this dissertation, and an survey of earlier works on the subject is listed.

Chapter 2, presents the fluid flow equations, and specifically the two-phase flow model that is used in this work.

Furthermore, in chapter 3 we explain how the fluid flow equations introduced in chapter 2 are discretised.

Chapter 4, states the production optimisation problem, which we are studying.

In chapter 5, we give a brief overview of inverse problems in general, and we introduce the problem of history matching. Also a section on regularisation is included.

In chapter 6, we discuss unconstrained minimisation, where we talk about line search methods, and give a overview of the unconstrained optimisation methods that is used in this work.

Chapter 7 continues to discuss constrained optimisation, where we state the optimality conditions. Moreover we explain what methods have been used to handle bounds on the variables and equality constraints.

Chapter 8 gives an explanation on what optimal control is. We state the optimal control problem as a saddle point problem of the Lagrangian and the augmented Lagrangian. Then we analyse different possible approaches for solving the problem, and motivates the methods used in this work.

Chapter 9 discusses future work, on the subject treated in this thesis.

Finally, chapter 10 summarises the papers that are included in part II of this thesis.

Part II:

Four papers are included in part The works included are the following,

Paper A: An Efficient Method for Smart Well Production Optimisation.

Daniel Chr. Doublet, Sigurd I. Aanonsen and Xue-Cheng Tai.
To be submitted to the Journal of Petroleum Science and Engineering

Paper B: Marching Schemes for the Augmented Lagrangian Method.

Daniel Chr. Doublet, Xue-Cheng Tai and Sigurd Ivar Aanonsen.
To be submitted to the SIAM Journal on Scientific Computing

Paper C:Efficient History Matching and Production Optimisation with the Augmented Lagrangian Method.

Daniel Chr. Doublet, Sigurd I. Aanonsen and Xue-Cheng Tai.
Submitted to SPE Journal
Presented at SPE Reservoir Simulation Symposium, 26-28 February 2007, Houston, Texas, U.S.A.

Paper D:Efficient Optimisation of Production from Smart Wells Based on the Augmented Lagrangian Method

Daniel Chr. Doublet, Raymond Martinsen, Sigurd I. Aanonsen and Xue-Cheng Tai.

Presented at 10^{th} European Conferance on the Mathematics of Oil Recovery.

Acknowledgements

We want to thank everyone that we have worked with during this project, and who have helped me completing this work.

I want to express my gratitude to my advisers Sigurd I. Aanonsen and Xue-Cheng Tai, for their support of my work and the interesting discussions, we have had during the three year period which this project has been done.

Also I want to thank Raymond Martinsen for writing the forward solver, which has been used in my work.

Furthermore I want to express my thanks to Jean-Charles Gilbert for several interesting discussions during my three months stay at INRIA Roqencourt in 2006. During my stay I learnt much about constrained optimisation, and was a great pleasure to learn from someone with so much knowledge of mathematics and constrained optimisation in particular.

I want to thank my colleagues at the Centre for Integrated Petroleum Research, for making it fun to go to work. Especially I want to thank Dmitry Eydinov for nice smoking breaks, and Bartek Vik for distractional conversations.

And I thank my wonderful wife, Flavia, for her support and encouragement during my work.

Daniel Christopher Doublet,
November 2007.

Contents

I Background **1**

1 Introduction **3**
1.1 Motivation 3
1.2 Problem 3
1.3 Literature Survey 5
1.3.1 Optimisation Techniques 5
1.3.2 Production Optimisation 5
1.3.3 History Matching 6
1.3.4 Augmented Lagrangian 7

2 Fluid flow in porous media **11**
2.1 Two-phase flow 11

3 The Forward model **15**

4 Production Optimization **19**
4.1 Smart well, production optimisation 19

5 Parameter estimation **23**
5.1 Inverse Problems 23
5.2 History matching 24
5.3 Regularisation 25

6 Unconstrained Optimisation **27**
6.1 Line Search Methods 27
6.2 Steepest Descent Method 28
6.3 Quasi Newton methods 28
6.3.1 The LBFGS method 30
6.4 Step Length 31
6.4.1 Wolfe conditions 32

7 Constrained Optimisation **33**
7.1 Constrained Optimisation Problem 33
7.2 First Order Optimality Conditions 34
7.3 Second Order Optimality Conditions 35
7.4 Inequality constraints 35
7.4.1 Bounds on the variables 35
7.4.2 Projected Gradients 36
7.5 Equality constraints 36
7.5.1 The additional Constraints on the Control Variables 37

8 Optimal Control **41**
8.1 Problem Formulation 41
8.2 The Saddle Points of L and L_c 43
8.3 Solving the Optimal Control Problem 45
8.4 The Adjoint Method 45
8.5 Augmented Lagrangian Methods 47
8.6 The Augmented Lagrangian Method 49
8.7 Marching Schemes 49
8.7.1 The new marching scheme 52
8.8 The KKT method 53
8.8.1 The role of the penalty constant in the KKT algorithm . . 56
8.9 Other Possibilities 56

9 Future Work **59**

10 Summary of Papers **61**
10.1 Summary of Paper A and D 61
10.2 Summary of Paper B 62
10.3 Summary of Paper C 63

Bibliography **65**

II Included Papers **71**

A An Efficient Method for Smart Well Production Optimisation. **73**

B Marching Schemes for the Augmented Lagrangian Method. **105**

C Efficient History Matching and Production Optimisation with the Augmented Lagrangian Method. **145**

D Efficient Optimisation of Production from Smart Wells Based on the Augmented Lagrangian Method 155

Part I

Background

Chapter 1

Introduction

1.1 Motivation

In connection with enhanced oil recovery, optimal control problems arise quite frequently. All parameter estimation problems or optimisation problems related to some sort of physical process can be formulated as an optimal control problem. Since, optimal control problems are so important in enhanced oil-recovery, it is seems to be a good idea to investigate the different methods we have for solving optimal control problems, and find which are suited for solving parameter estimation problems in connection with oil recovery. Work has earlier been done, with the so called adjoint method for these type of problems, but the use of augmented Lagrangian methods has not to our knowledge been applied to problems occurring in connection with enhanced oil-recovery. Although the adjoint methods has proved to be successful, it is always of interest to improve the existing technology, so that we can solve these problems faster and more accurately. In this work, we attempt to investigate the use of the method of multipliers for optimal control, with the application for production optimisation with water flooding and for history matching. Additionally, we investigate other possible methods, using the augmented Lagrangian functional. Intuitively it appears to be a good idea to use this sort of formulation, since in the adjoint method one always have to solve the forward problem at each iteration, which is the most expensive

1.2 Problem

Water flooding is frequently used as a means to enhance oil recovery, and due to heterogeneous permeability one may experience early water breakthrough resulting in oil being trapped in the reservoir. However, by controlling the injection and production in the wells we may avoid or minimize this problem.

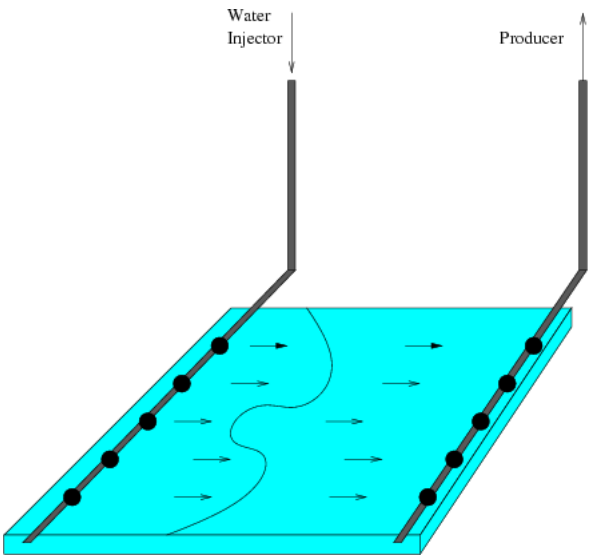

Figure 1.1: Water flooding of reservoir with smart wells. The figure is inspired by Brouwer et al. [7]

Smart wells are wells which have sensors and valves on the tubing, allowing us to control the injection/production at the separate valve openings. Figure 1.1 shows a horizontal reservoir with one smart well injector and one smart well producer.

Optimising of the profit or oil recovery from an oil reservoir, can be achieved through controlling the injection/production in the different wells. In the case of smart wells, one has in addition the possibility to control the injection/production at each valve opening of the well which allows for a better flooding of the wells.

One can then control the production/injection in such a way that one minimises the amount of trapped oil in the reservoir, and thereby maximising the net present value(NPV) from the reservoir.

In order to maximise the production, we need a reservoir simulator and thus a reservoir description (permeabilities, porosities etc.).

At the beginning of production we have some initial parameters, which describe the reservoir and we can use this for optimising the production. Once the optimisation has terminated, we continue to produce according to the optimal production plan. While producing the wells continuously provide measurement data, which can be used to history match the geological parameters. When a new update of the geological parameters are available the production optimisation is then done again. A schematic view of this process can found in figure 1.2. Since the optimisation problem must be done repeatedly, it is therefore important to do

have fast methods to do both the production optimisation and to solve the history matching problem.

1.3 Literature Survey

1.3.1 Optimisation Techniques

Optimisation techniques can be divided into two groups of methods -stochastic methods and deterministic methods. Stochastic methods, such as genetic algorithms [45], [25] and simulated annealing [34], [45],[39] make only use of function evaluation and does not include gradient information. These methods has, usually much slower convergence rate than deterministic approaches, which include gradient information when searching for the optimum. These methods include for example the Steepest Descent method, the non-linear conjugate gradient method, quasi-Newton methods, the Gauss-Newton method and Levenberg-Marquardt. The deterministic, gradient based methods generally converge faster, and in this work we have only considered these type of methods.

1.3.2 Production Optimisation

We divide production optimisation into two groups, reactive and predictive approach. Reactive methods are methods that change valve settings when change in water production occurs, such that regions producing too much water are shut down.

In Yeten et al. [47] and Kharghoria et al. [31], genetic algorithms are used to optimise production by reducing production in zones with high water cut. Brouwer et al. [8] consider water flooding with smart wells, where the production is decreased or shut in when water breakthrough occurs. In this case the production strategy is also a direct reaction to the measurements from the valve openings. Gai [18] investigate the optimisation of valve settings for a multilateral well, where the decisions are based on well measurements, and this comes into the group of reactive control. In the work by Arenas and Dolle [2], a two-dimensional reservoir with fractures where valves are only present in the injector. The valves are either fully open or closed, and they are open or closed as a response to water cut measurements in the producer. Snaith et al. [46] documents the use of reactive control has to real reservoirs, and Al-Khodhori [1] also shows the use of reactive optimisation in a real application.

When using reactive control, no action is performed before some change in the well measurements is observed. The profit of the reservoir can be enhanced even further if one acts before. If for example, change in valve settings is done when

water breakthrough occurs, we might then already have large volumes of trapped oil in the reservoir. Thus, by using reservoir simulators one can predict the fluid flow in the reservoir and hence optimise it by adjusting the valve settings. In this way, the optimal valve settings are found before the measurements are made, and this strategy can be used to avoid possible problems occurring at the wells at a later time. This is what is termed predictive control.

Brouwer et al. [7] uses predictive control, where they use an optimal control theory method, the adjoint method, in combination with a gradient based optimisation algorithm, to find the optimal valve settings. Horizontal, two-dimensional reservoirs are considered, and they maximise the net present value (NPV), where they associate a cost with water production and a profit with oil production. The approach has provided good results, but it has the disadvantage that one has to solve the forward problem each time a new estimate of the controls is available.

Lien et al. [33] use the same problem definition as in Brouwer et al. [7], where the adjoint method is used in combination with a multiscale regularisation. The results are good and, with the aid of multiscale regularisation, the adjoint method converges faster and in some examples they find a higher profit than one does with the adjoint method alone. Additionally there has been done some work by Zakirov et al. [48] and Sarma et al. [44], using the adjoint method for optimising the production.

In this work we study the use of optimal control techniques to optimise net present value by adjusting the valve settings in smart wells. We take the problem definition from Brouwer et al. [7]. Our aim is to investigate the use of augmented Lagrangian methods, to solve the optimal control problem. In augmented Lagrangian methods, there will not be a need to solve the forward problem every time a new estimate of the controls is available, and consequently one iteration of such an algorithm is cheaper than one iteration of the adjoint method. The forward model fulfils itself at the same time as the NPV is maximised as the algorithms converge. Due to this, it has been of interest to see if one can devise algorithms, based on the augmented Lagrangian method, which are cheaper in terms of computational effort, than the adjoint method.

1.3.3 History Matching

The history matching problem is similar to that of the production optimisation problem, since they both can be regarded as optimal control problems. However, the history matching problem is an ill-posed problem which present supplementary complications, compared to that of the production optimisation problem. On the other hand, for the history matching problem, the objective function has the form of a least-squares problem, while the production optimisation problem does not.

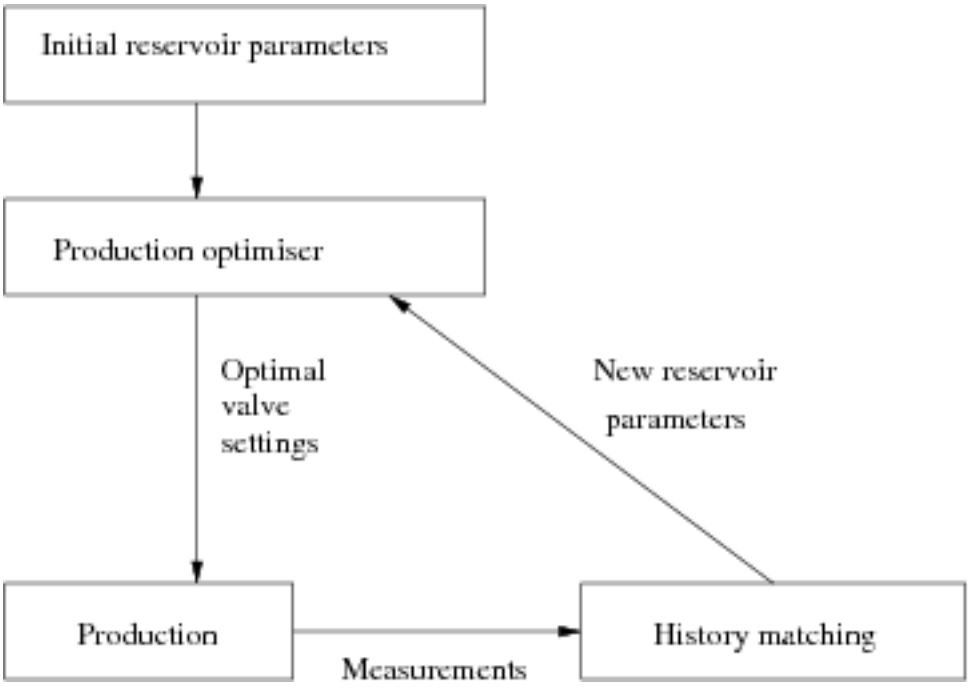

Figure 1.2: Schematic view of production optimisation cycle.

There has been done many studies of the history matching problem, and optimal control methods have been used to address this problem as well. Chen et al. [14] and Chavent et al. [13] stated the problem as an optimal control problem and used an adjoint algorithm as solution method, and later others has continued the use of adjoint methods for the solution of the problem, see for example Ramirez [43].

As for the production optimisation problem, we need to solve the forward problem every time we have a new set of parameters available. Therefore we want to investigate the use of augmented Lagrangian methods to solve the history matching as well, although the production optimisation problem is the main problem considered here.

1.3.4 Augmented Lagrangian

The augmented Lagrangian method was first introduced independently by Hestenes [24] and Powell [42]. Hestenes considered the method for solving constrained optimisation problems. Glowinski [19], [20] has studied the augmented Lagrangian method for non-linear problems, and Bertsekas [5] has also studied the method for solving constrained optimisation problems, and has studied the

possibility of using second order methods in combination with the augmented Lagrangian methodology [4].

Later the use of the augmented Lagrangian method was studied by Itô and Kunisch [27, 28, 26] where they considered the use of the augmented Lagrangian method for parameter identification in elliptic systems. They considered a least-squares objective function with regularisation term.

Kunisch and Tai [32], considered a more general optimal control problem, where the study the parameter identification problems, where the controls and the state variables are related through a bilinear term and they present numerical examples

Chan and Tai [11] used the augmented Lagrangian method with total variations regularisation for finding discontinuous parameters in elliptic and hyperbolic systems. Chen et al. [15] also used an augmented Lagrangian approach to identify discontinuous parameters in elliptic problems.

Keung and Zou [30, 29], also considered parameter identification in parabolic and in elliptic systems, where they in the latter paper used a modified Uzawa algorithm of the augmented Lagrangian method to solve the problem.

Guo and Zou [21] also has investigated the use of the augmented Lagrangian method to identify parameters in parabolic systems.

Nilssen and Tai [36] considered the use of marching schemes of the augmented Lagrangian method to solve parameter identification problems from a parabolic system, and and Nilssen et al. [37] considered the same marching schemes for estimating permeability from a non-linear diffusion equation.

The goal of this work, is to investigate how augmented Lagrangian methods can be used to solve the production optimisation problem, and other optimal control problems that occur in connection with reservoir engineering. The studies done with the augmented Lagrangian method for solving optimal control problems, has only(to our knowledge) been focused on parameter identification problems.

These type of problems have a least-squares objective function, where the goal is to find the best fit to some measurements. In this study, we investigate an optimal control problem where the objective function is not of the type least-squares. The difference is that we do not know the value of the objective function at the optimum.

In this study we have investigated the marching schemes of the augmented Lagrangian used in [36, 37], and investigated if they can be used to solve the production optimisation problem.

Our studies indicate that the marching schemes in [36, 37], are not well suited for problems where the objective function is not on the form of a least squares function, and we have proposed a new marching scheme that circumvents this problem. Numerical examples show that it converges, and the new marching

scheme is a method of applying the augmented Lagrangian method to optimal control problems, which can be used with all sorts of objective functions.

Based on an augmented Lagrangian saddle point problem formulation of the optimal control problem, we have proposed a new method to solve optimal control problems, which is called the KKT method. With this method we have used solved production optimisation problems and history matching problems with this method, and it shows rapid convergence to a high value of the NPV.

Chapter 2

Fluid flow in porous media

Since our work is concentrated on solving optimal control problems, which occur in connection to reservoir engineering, we will give a general survey of modelling of fluid flow in porous media.

The porous medium is a rock which contains a network of pores. In petroleum exploitation, we are considering rocks in which fluids can flow. Thus we are considering rocks where the pores are connected, so that it is possible to have fluid flow through the medium. The porous medium is characterised by its porosity ϕ and its absolute permeability κ, which both may vary with the position in the media. The porosity is the percent of volume of pores of the medium, and it is defined by

$$\phi = \frac{V_p}{V_T}, \tag{2.1}$$

where V_p is the volume of the connected pores, where fluids can flow, and V_T is the total volume. The absolute permeability is a measurement of how well a fluid flows through the medium, and it is dependent on the position. In this work, we have only considered two-phase fluid flow (oil and water), and we give here a short description of the fluid flow equations that are used in this work. For more information on fluid flow in porous media, the reader may consult for example [3], [40], [23], [41].

2.1 Two-phase flow

We will now consider the case where we have two fluids present in a porous media, where there is no mass transfer between the two phases. This is a situation that occurs when we displace oil by water in the reservoir. This situation occurs often in practice, since the water is commonly injected in to reservoirs in order to facilitate oil production. First we must introduce some new variables in order

to describe the fluid flow, and we will here denote the two different phases by $\alpha = \{o, w\}$, where o means water and w means water. The saturation of phase α, S_α is defined as the volume of the fluid α divided by the total pore volume, that is

$$S_\alpha = \frac{V_i}{V_p}. \tag{2.2}$$

Furthermore it is natural to assume that the porous media is completely saturated, and thus we must have that

$$S_o + S_w = 1. \tag{2.3}$$

Both the saturation's for water and oil, must be between zero and one, but the saturation for phase α cannot be below the what is called the residual saturation for phase α, $S_{\alpha r}$, meaning that the reservoir can never be completely saturated with one of the fluids. Thus we have that

$$S_{wr} \leq S_w \leq 1 - S_{or} \tag{2.4}$$

$$S_{or} \leq S_o \leq 1 - S_{wr}. \tag{2.5}$$

For each of the phases mass is conserved, and we will for simplicity consider the conservation of mass per unit volume. Given a reference volume of the porous medium, Ω, the conservation of mass for phase α in Ω is given by

$$\int_\Omega \frac{\partial \Gamma_\alpha}{\partial t} dV = -\int_{\partial\Omega} F ds + \int_\Omega q dV, \tag{2.6}$$

where Γ_α is the mass of phase α per unit volume, F is the flux of Γ_α over the boundary $\partial\Omega$ of Ω and q is source and sink terms within Ω. Using the divergence theorem we find that

$$\int_\Omega \frac{\partial \Gamma_\alpha}{\partial t} dV = -\int_\Omega \nabla \cdot F dV + \int_\Omega q dV. \tag{2.7}$$

Since the equation (2.7) is valid for any arbitrary volume Ω, equation (2.7) must also be valid point wise, giving

$$\frac{\partial \Gamma_\alpha}{\partial t} = -\nabla \cdot F + q. \tag{2.8}$$

The mass per unit volume is given as

$$\Gamma = \phi \rho_\alpha S_\alpha, \tag{2.9}$$

where ρ_α is the density of phase α. Furthermore, the flux is given as the velocity times the density,

$$F = \rho_\alpha u_\alpha. \tag{2.10}$$

The velocity u is described by Darcy's law,

$$u_\alpha = -\frac{\kappa_\alpha}{\mu_\alpha}(\nabla p_\alpha + \rho_\alpha g h), \tag{2.11}$$

where g is the gravitational acceleration, h is height, μ_α is the viscosity and κ_α is the effective permeability for fluid α. Since two fluids are present in the porous medium, the effective permeability is lesser than the permeability when there is only one fluid present in the medium (the absolute permeability), since the mobility of each fluid is hindered by the presence of the other. The effective permeability is the absolute permeability, κ, times the relative permeability $\kappa_{r\alpha} = \kappa_{r\alpha}(S_\alpha)$ which is a scalar between 0 and 1, so that the effective permeability is given by

$$\kappa_\alpha = \kappa\kappa_{r\alpha}(S_\alpha). \tag{2.12}$$

By combining the mass conservation equations (2.8) and (2.11) we find that

$$\frac{\partial \phi \rho_o S_o}{\partial t} - \nabla \cdot \left(\rho_o \kappa \lambda_o (\nabla p_o + \rho_o g h)\right) = q_o, \tag{2.13}$$

$$\frac{\partial \phi \rho_w S_w}{\partial t} - \nabla \cdot \left(\rho_w \kappa \lambda_w (\nabla p_w + \rho_w g h)\right) = q_w, \tag{2.14}$$

where the mobility $\lambda_\alpha = \frac{\kappa_{r\alpha}}{\mu_\alpha}$. There will be a difference in the pressure for the two fluids, and we call this for the capillary pressure, P_c, defined by

$$P_c(S_w) = p_o - p_w, \tag{2.15}$$

where water is assumed to be wetting the medium relatively to the oil. The two-phase flow of water and oil a porous medium is now described completely by the equations (2.3),(2.13),(2.14) and (2.15). These equations are referred to as the fluid-flow equations for two-phase flow in porous media.

Chapter 3

The Forward model

In order to model our reservoir flow equations we make use of an in-house reservoir simulator for two-phase (water and oil), two-dimensional flow where we ignore gravitational effects and capillary forces, and we will here give a description of the model. The reservoir flow equations can be modelled in many different ways, but we have chosen a simple model since the aim our study is the solution of optimal control problems in connection with reservoir flow, and not to find the optimal reservoir model. For discussions on how to do forward modeling, we refer the reader to for example Aziz and Sattari [3], Ewing [16], Chavent and Jaffré [12] or Heimsumd [23].

Since the capillary pressure is zero, we will denote the pressure as p. In order to discretise the flow equations (2.13) and (2.14), we use a cell centred finite difference method, where the backward Euler method is used to discretise the time derivative. We divide our reservoir into a discrete number of grid cells as shown in figure 3.1, where we assume that all variables are constant within each cell. The horizontal axis is divided into n_x intervals, and the vertical axis is divided into n_y intervals, giving a total of $n_x \times n_y$ grid cells. The length of the vertical edge of cell $\{i, j\}$ is Δy_j and its horizontal length is Δx_i. Furthermore we divide the time line into a discrete set of N time intervals, where the length of time interval number n is denoted by Δt^n. The interval corresponding to Δt^n is called time step n. We want that our equations should be on dimensionless form, and therefore we multiply by the discrete time step Δt^n and we find that the discrete approximation of the flow equations for cell $\{i, j\}$ and phase $\alpha \in \{w, o\}$ at time step n is given by

$$\phi_{i,j}(S^n_{i,j,\alpha} - S^{n-1}_{i,j,\alpha}) - \Delta t^n \Phi^n_{i,j,\alpha} = \Delta t^n q^n_{i,j,\alpha}, \tag{3.1}$$

where the operator $\Phi^n_{i,j,\alpha}$ is a discrete approximation to the second term in equations (2.13) and (2.14). The discrete operator $\Phi^n_{i,j,\alpha}$ is discretised according to a

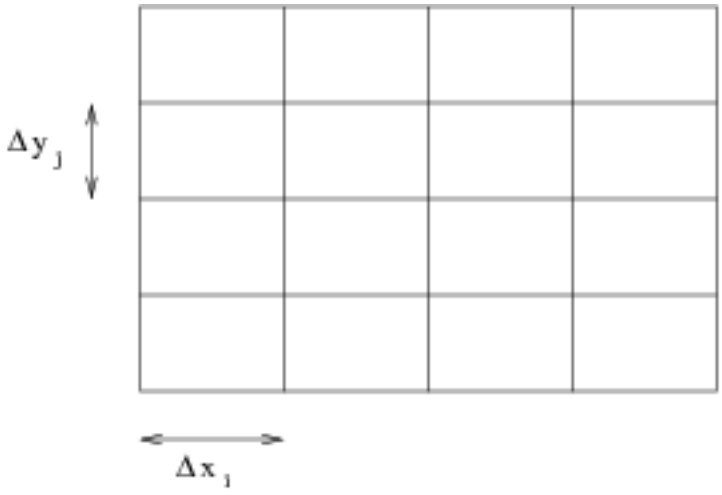

Figure 3.1: Grid

formula from [3]. When omitting the subscript α, it gives that

$$\Phi^n_{i,j} = \left(\lambda_{i+\frac{1}{2},j} T_{i+\frac{1}{2},j}(p^n_{i+1,j} - p^n_{i,j}) - \lambda_{i-\frac{1}{2},j} T_{i-\frac{1}{2},j}(p^n_{i,j} - p^n_{i-1,j})\right)/\Delta x_i \tag{3.2}$$

$$+ \left(\lambda_{i,j+\frac{1}{2}} T_{i,j+\frac{1}{2}}(p^n_{i,j+1} - p^n_{i,j}) - \lambda_{i,j-\frac{1}{2}} T_{i,j-\frac{1}{2}}(p^n_{i,j} - p^n_{i,j-1})\right)/\Delta y_j, \tag{3.3}$$

where the transmisibilities T are given by

$$T_{i+\frac{1}{2},j} = \frac{1}{\frac{1}{2}\left(\frac{\Delta x_i}{\kappa_{i,j}} + \frac{\Delta x_{i+1}}{\kappa_{i+1,j}}\right)}, \tag{3.4}$$

$$T_{i-\frac{1}{2},j} = \frac{1}{\frac{1}{2}\left(\frac{\Delta x_i}{\kappa_{i,j}} + \frac{\Delta x_{i-1}}{\kappa_{i-1,j}}\right)}, \tag{3.5}$$

$$T_{i,j+\frac{1}{2}} = \frac{1}{\frac{1}{2}\left(\frac{\Delta y_j}{\kappa_{i,j}} + \frac{\Delta y_{j+1}}{\kappa_{i,j+1}}\right)}, \tag{3.6}$$

$$T_{i,j-\frac{1}{2}} = \frac{1}{\frac{1}{2}\left(\frac{\Delta y_j}{\kappa_{i,j}} + \frac{\Delta y_{j-1}}{\kappa_{i,j-1}}\right)}. \tag{3.7}$$

For the mobilities λ, we use upstream weighting. That is

$$\lambda_{i+\frac{1}{2},j} = \begin{cases} \lambda_{i+1,j}, & p_{i+1,j} > p_{i,j} \\ \lambda_{i,j}, & p_{i+1,j} \leq p_{i,j} \end{cases} \tag{3.8}$$

$$\lambda_{i-\frac{1}{2},j} = \begin{cases} \lambda_{i-1,j}, & p_{i-1,j} > p_{i,j} \\ \lambda_{i,j}, & p_{i-1,j} \leq p_{i,j} \end{cases} \tag{3.9}$$

$$\lambda_{i,j+\frac{1}{2}} = \begin{cases} \lambda_{i,j+1}, & p_{i,j+1} > p_{i,j} \\ \lambda_{i,j}, & p_{i,j+1} \leq p_{i,j} \end{cases} \tag{3.10}$$

$$\lambda_{i,j-\frac{1}{2}} = \begin{cases} \lambda_{i,j-1}, & p_{i,j-1} > p_{i,j} \\ \lambda_{i,j}, & p_{i,j-1} \leq p_{i,j} \end{cases} \tag{3.11}$$

For the relative mobilities, we will use the Corey models, i.e.

$$\kappa_{ro} = \kappa_{ro}^{\star}\Big(\frac{1-S-S_{or}}{1-S_{or}-S_{wr}}\Big)^{e_o} \tag{3.12}$$

and

$$\kappa_{rw} = \kappa_{rw}^{\star}\Big(\frac{S-S_{wr}}{1-S_{wr}-S_{or}}\Big)^{e_w}, \tag{3.13}$$

where e_o and e_w are the Corey exponents, $\kappa_{ro}^{\star}$ and $\kappa_{rw}^{\star}$ are the endpoint permeabilities, and S_{or} and S_{wr} are the residual saturations, for oil and water respectively.

Furthermore, we assume that the liquids are incompressible, and let the total injection/production rate, V, be constant, not varying with time. We will assume that we have two smart wells, one injector and one producer, where we can control the injection/production in the different valve openings along the two wells. We denote the number of valve openings along the injector by N_{inj} and the number of valve openings along the producer by N_{prod}. Additionally, we define the percentage of the total injection/production in well segment i at time step n as v_i^n. Since the v_i^n are percentages, we have that, for $n = 1, \ldots, N$

$$0 \leq v_i^n \leq 1 \quad \text{for} \quad i = 1, \ldots, N_{inj} + N_{prod}. \tag{3.14}$$

Moreover, the sum of the injection rates over all the valve openings along the injector must equal the total injection, and for the producer the sum of the production rates over all the valve openings must also equal the total production rate V. This gives us the additional equality constraints for $n = 1, \ldots, N$

$$\sum_{i=1}^{N_{inj}} v_i^n = 1, \tag{3.15}$$

and

$$\sum_{i=N_{inj}+1}^{N_{inj}+N_{prod}} v_i^n = 1. \tag{3.16}$$

Since we only inject water, the injection rate at well segment i is

$$q_{wi}^n = v_i^n V, \quad \text{for} \quad i = 1, \ldots, N_{inj}, \tag{3.17}$$

where q_{wi}^n is the water rate at injection segment i at time step n. The different phase production rates can be expressed as functions of the liquid rate and the fractional flow at the well segment so that

$$q_{wi}^n = -\frac{\lambda_{wi}^n}{\lambda_{wi}^n + \lambda_{oi}^n} \, v_i^n V \quad \text{for} \quad i = 1 + N_{inj}, \ldots, N_{prod} + N_{inj}, \tag{3.18}$$

and

$$q_{oi}^n = -\frac{\lambda_{oi}^n}{\lambda_{oi}^n + \lambda_{wi}^n} \, v_i^n V, \quad \text{for} \quad i = 1 + N_{inj}, \ldots, N_{prod} + N_{inj}, \tag{3.19}$$

where λ_{wi}^n and λ_{oi}^n are the water and oil mobilities, respectively, in the grid cell containing well segment i and at time step n. The discrete approximations to the reservoir flow equations for time step n (3.1) can be written on vectorial form as

$$e^n(v^n, u^n, u^{n-1}) = \left\{ \begin{array}{c} e_o^n \\ e_w^n \end{array} \right\} = 0 \tag{3.20}$$

where the e^n is the discrete equation residual, the subscripts w and o denotes the equation residual for the flow equations for oil and water, respectively, and where $u^n = \{p^n, S_w^n\}$. The pressures and the water saturations, grouped together as $u = \{u^n\}_{n=1}^N$, are referred to as the state variables. Now we have that

$$e_\alpha^n(v^n, u^n, u^{n-1}) = \phi(S_\alpha^n - S_\alpha^{n-1}) - \Delta t^n \Phi_\alpha^n - \Delta t^n q_\alpha^n, \tag{3.21}$$

where this vector equation is organised so that the flow equation for grid cell $\{1, 1\}$ is its first element, grid cell $\{n_x, 1\}$ is element n_x in the vector and grid cell $\{n_x, n_y\}$ is the last element.

Chapter 4

Production Optimization

In this chapter we present the problem of maximising the profit from an oil-reservoir by controlling the injection/production and we formulate the problem as an optimal control problem.

4.1 Smart well, production optimisation

When an reservoir goes in to the secondary recovery phase, water is injected in order to enhance oil-recovery. Since the rock is a heterogeneous medium, water will flow with different velocities throughout the reservoir. Due to this, water breakthrough can occur when great regions of the reservoir are still unflooded so that much of the oil may become ”trapped” in the reservoir. Such a situation might make it necessary to drill additional wells in order to recover the remaining oil, which is a costly procedure along with the halt of production.

Smart wells with down-hole valves gives us the possibility to control the injection/production at each of the valve openings along the well, so that it is possible to control the flow-regime. A schematic figure is depicted in figure 1.1. One can control the injection/production at all valve openings, and this setting of the valves may be changed during the production period, which gives us a great deal of control over the production and we want to control the injection/production so that the profit obtained from the reservoir is maximised.

Increased recovery rate and reduced water production increases the profit from the reservoir, since it costs money to remove water from the oil. Additionally, one wants to produce the reservoir as fast as possible, in order to reduce the total operating costs.

The production optimisation problem is often formulated as that of maximis-

ing the net present value (NPV), given by

$$J_{NPV} = \sum_{n=1}^{N} J_{NPV}^{n}, \tag{4.1}$$

where J_{NPV}^{n} is the NPV obtained during the time interval Δt^n, which is

$$J_{NPV}^{n} = \Delta x \Delta y h \left[\sum_{i=1+N_{inj}}^{N_{prod}+N_{inj}} \frac{-I_w \cdot q_{wi}^{n} - I_o \cdot q_{oi}^{n}}{(1+b/100)^{t^n}} \right] \Delta t^n, \tag{4.2}$$

where the constants I_o and I_w are, respectively, the revenue of oil produced and the cost of water produced per volume expressed in $\$/m^3$, b is the annual interest rate expressed in %, Δx and Δy are the dimensions of the grid cells in, respectively, horizontal and vertical direction(using a uniform grid), Δt^n is the size of the n'th time step, and $t^n = \sum_{i=1}^{n} \Delta t^i$ is the time expressed in years at time step n. Since the production rates of water and oil, q_{wi}^{n} and q_{oi}^{n}, are less than or equal to zero, I_o is a positive constant and I_w is a negative constant.

For our controls to physically feasible, the reservoir model must be fulfilled, and our problem is therefore an optimal control problem, which we can formulate as follows

$$\max_{v} J(v,u) \quad \text{subject to} \quad e^n(v^n, u^n, u^{n-1}) = 0, \quad \text{for} \quad n = 1, \ldots N, \tag{4.3}$$

where v are our control variables and u are our state variables. The state variables u are the pressures and the saturations in all the grid cells and for the controls, v, we use the percentage of total injection/production in each of the valve openings along the injector and the producer.

In practice, the injection/production is controlled at each valve opening by adjusting the surface of the valve opening, but since this can easily be converted to pressure control or injection/production rate control and we assume therefore, for simplicity, that we may control the injection/production rate directly.

In addition to the state equations constraints, we also have bounds on the controls and the state variables, and also the equality constraints (3.15) and (3.16), and we restate here these additional constraints. As earlier defined, v_i^n denote the percent of total injection or production in valve opening i at time step n, and the constraints are

$$\sum_{i=1}^{N_{inj}} v_i^n = 1, \quad \text{for} \quad n = 1, \ldots, N, \tag{4.4}$$

and

$$\sum_{i=1+N_{inj}}^{N_{prod}+N_{inj}} v_i^n = 1, \quad \text{for} \quad n = 1, \ldots, N. \tag{4.5}$$

All the valve openings from number 1 to number N_{inj} are assumed to belong to the injection well, while the valve openings from number $1 + N_{inj}$ to number $N_{prod} + N_{inj}$ are assumed to belong to the production well. Moreover, the control variables must also satisfy the following inequality constraints, for $n = 1, \ldots, N$,

$$0 \leq v_i^n \leq 1, \quad \text{for} \quad i = 1, \ldots, N_{inj} + N_{prod}. \tag{4.6}$$

Furthermore for the state variables, $u = \{p, S\}$ we have that

$$S_{wr} \leq S \leq 1 - S_{or} \tag{4.7}$$

and that

$$p > 0. \tag{4.8}$$

Now the equations (4.3), (4.4), (4.5), (4.6), (4.7) and (4.8) define our production optimisation problem. In the following chapters we will specify how this problem is handled.

Chapter 5

Parameter estimation

When simulating the fluid flow in a reservoir, we need to have information on the characteristics of the reservoir, such as absolute and relative permeability, porosity and capillary pressure in order to simulate accurately. These parameters cannot be observed directly since we have no way of 'looking' into the reservoir and we must thus try to estimate their value via some indirect approach. The parameter estimation problems are problems of a class of problems called inverse problems.

5.1 Inverse Problems

Inverse problems can generally be written as

$$G(s) = b, \tag{5.1}$$

where G is some operator which work on s and give b. In the direct problem, G and s are known, and one seeks to determine b. The inverse problem is when either b and G is known, and one seeks s or when b and s is known and one seeks to find G. Parameter estimation problems in reservoir mechanics, is of the first type. These type of problems are typically more difficult than their direct counterpart, and the difficulties are usually as a result of ill-posedness, which we define next. The mathematician Hademard [22] defined a problem to be well posed if all of the three conditions hold

Well Posedness

1. The solution exists,
2. The solution is unique,
3. The solution depends continuously on the data.

Problems where one of the above conditions is not fulfilled, are called ill-posed problems. For parameter estimation problems in reservoir engineering context, we have some measurements of pressure, water-cut and other quantities in the wells, in addition to data obtained from seismic surveys. The task is then to find the unknown parameters which best fit the production history data and the data obtained from other sources such as seismic surveys etc. Often these problems are referred to as history matching, since one tries to match the production data.

5.2 History matching

In this work, we estimate the absolute permeability The measurements which we use, consist of two types of measurements where the first is measurements of pressures and saturations in both the production and injection wells. Even though the saturations are not measured directly from a real-life field, they can easily be found and we have thus treated them as measurements. Additionally we have at certain distinct time steps, measurements seismic surveys available giving us information of the pressure and saturations in the entire reservoir. The inversion of seismic data, is an inverse problem but for simplicity we have assumed that the seismic gives pressure and saturation at every grid cell at certain time steps. This leaves us with essentially two types of measurement data, where the well-measurements are made often they are made at few points of the reservoir (the wells). On the other hand, the seismic measurements are made more seldom, but they provide information over the entire reservoir. This problem does not have a unique solution and violates thus the second condition of the well-posed criteria. We will denote data measurements as d_{obs}, and we will denote the data obtained from the reservoir model given a permeability field κ for

$$d_{sim} = g(\kappa), \tag{5.2}$$

where g is the forward model operator, which gives the pressures and saturations at the same instances and locations as the measurements. Now our goal is to find κ so that

$$d_{sim} - d_{obs} \tag{5.3}$$

is minimal. To find a reasonable objective function, we ask what the probability is to find d_{obs} given the parameters κ. The probability density function is then,

$$P(d_{obs}|\kappa) = a \cdot exp\big[-\frac{1}{2}(d_{sim} - d_{obs})^T C_D^{-1}(d_{sim} - d_{obs})\big], \tag{5.4}$$

where a is a constant scalar and C_D is the covariance matrix of the measurements, containing information about the uncertainties of the observations d_{obs}. Now we

want to find κ so that the probability in (5.4) is maximised, since its maximum will give the most probable distribution that resulted in the measurements d_{obs}. Maximising of (5.4) is equivalent to the problem of minimising the function

$$J_{HMo} = \frac{1}{2}(d_{sim} - d_{obs})^T C_D^{-1}(d_{sim} - d_{obs}). \tag{5.5}$$

5.3 Regularisation

Due to the ill-posedness of the problem, we cannot know if the minimum of our problem (5.5) will give us the solution which we desire. If the problem has a non-unique solution it makes sense to try to incorporate some prior information, so that the we find a solution which agrees with our prior known information. One source of prior information, may for example be some geostatistical model of the permeability, κ_{prior}, with a corresponding covariance matrix, C_P, containing information about the uncertainties of this prior model. We may then form the new objective function as

$$J_{HM} = \frac{1}{2}(d_{sim} - d_{obs})^T C_D^{-1}(d_{sim} - d_{obs}) + \frac{1}{2}(\kappa - \kappa_{prior})^T C_M^{-1}(\kappa - \kappa_{prior}), \tag{5.6}$$

which is the data matching objective function (5.4) plus the prior term.

Chapter 6

Unconstrained Optimisation

Unconstrained optimisation is the simplest case of optimisation, where one wants to minimise some function $J : \mathbb{R}^{N_x} \to \mathbb{R}$,

$$\min_x J(x). \tag{6.1}$$

Methods for solving (6.1), can divided into two groups of optimisation methods, line search methods and trust region methods. Since we have only considered line search methods in our work, we will not discuss trust region methods here. For more information on this topic, one can consult for example Bonnans et al. [6] or Nocedal and Wright [38].

6.1 Line Search Methods

Optimisation algorithms where we solve problems of the form (6.1), usually finds a sequence of variables $x_1, x_2, \ldots$ such that

$$\lim_{k\to\infty} J(x_k) = \min_x J(x). \tag{6.2}$$

The sequence of variables are found by first choosing an initial value x_0 and then updating for $k = 1, 2, \ldots$

$$x_{k+1} = x_k + \alpha_k p_k, \tag{6.3}$$

where p_k is the search direction and α_k is a positive scalar called the step length. If x_k is to be a minimising sequence, we must have that p_k is a descent direction, i.e.

$$(\nabla_x J(x_k), p_k) < 0. \tag{6.4}$$

Once the search direction p_k has been determined, the step length α_k can then be found from the following minimisation problem

$$\min_{\alpha_k} J(x_k + \alpha_k p_k). \tag{6.5}$$

Firstly we will discuss some methods for calculating the search direction, and then we will discuss how to find the step length α_k.

6.2 Steepest Descent Method

The simplest line search method for solving unconstrained minimisation problems is the steepest descent method, where the search direction equals the negative of the gradient, that is

$$p_k = -\nabla_x J(x). \tag{6.6}$$

We see that this search direction is a descent direction c.f. (6.4), but the method may be slowly convergent. Consequently we we have chosen to use a second order method to solve our optimisation problems, since these methods are known to converge faster.

6.3 Quasi Newton methods

Second order methods make use of the Hessian to calculate the search direction, but the Hessian is not always easily accessible, i.e. it may be computationally expensive to find it, and we can instead use some approximation of the Hessian. Second order methods, where the Hessian is approximated are called Quasi-Newton methods. Like the steepest descent method, these type of methods only require evaluation of the objective function and its gradient. But information from previous iterations is used to approximate the Hessian, and the resulting convergence rate is much better than for the steepest descent method. For quasi-Newton methods, we start with making the following quadratic model, m, of the objective function, J at the current estimate x_k, giving

$$J(x_k + p_k) \approx m(p_k) = J(x_k) + \nabla J^T(x_k)p_k + \frac{1}{2}p_k^T B_k p_k, \tag{6.7}$$

where B_k is an $n \times n$ symmetric positive definite matrix which approximates the second derivative. The minimiser, p_k, of this quadratic model (6.7) is given as

$$p_k = -B_k^{-1}\nabla J(x_k). \tag{6.8}$$

If we use the true Hessian, for B_k in (6.8), the search direction becomes the Newton search direction. However, the idea of quasi-newton methods is to use some sort of approximation to the Hessian in (6.8) and use the resulting search direction in the line search method. Since the meaning of using an approximation to the Hessian, instead of calculating it is to reduce the computational effort spent, it

is important that our Hessian approximation can be found easily, and we wish to have some sort of update of B_k. In addition to requiring that, B_k, should be symmetric positive definite for all k, we demand that the gradient of m should match the gradient of J at the two latest iterates, x_{k-1} and x_k. Clearly, $\nabla m(0) = \nabla J(x_k)$, and for the $k-1$ iterate, we have that

$$\nabla m(p_{k-1}) = \nabla J(x_k) - B_k p_{k-1} = \nabla J(x_{k-1}), \tag{6.9}$$

which can be written as

$$B_k s_{k-1} = y_{k-1}, \tag{6.10}$$

where

$$s_k = x_{k+1} - x_k, \tag{6.11}$$

and

$$y_k = \nabla J(x_{k+1}) - \nabla J(x_k). \tag{6.12}$$

This is called the quasi-Newton or the secant equation. In order to compute the search direction in (6.8), we need to find the inverse of B_k, thus it is an advantage if we are able to update the inverse of the Hessian, instead of solving a linear system of equations every time we need a new search direction. Letting $H_k = B_k^{-1}$, the secant equation for H_k becomes

$$H_{k+1} y_k = s_k. \tag{6.13}$$

The update of B_k and H_k, so that they satisfy the secant equation and so that B_k and H_k are symmetric positive definite can be done in large number of ways. The first quasi-Newton method was the method of Davidon-Fletcher-Powell (DFP), where

DFP

$$B_{k+1} = \left(I - \gamma_k y_k s_k^T\right) B_k \left(I - \gamma_k s_k y_k^T\right) + \gamma_k y_k y_k^T, \tag{6.14}$$

with the corresponding update of the inverse Hessian,

$$H_{k+1} = H_k - \frac{H_k y_k y_k^T H_k}{y_k^T H_k y_k} + \frac{s_k s_k^T}{y_k^T s_k}. \tag{6.15}$$

One of the most common quasi-Newton methods is the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method, which we have chosen to use. In this method, B_k is updated by

BFGS

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}. \tag{6.16}$$

with the corresponding update of the inverse Hessian,

$$H_{k+1} = \left(I - \rho_k y_k s_k^T\right) H_k \left(I - \rho_k s_k y_k^T\right) + \rho_k y_k y_k^T, \tag{6.17}$$

where

$$\rho_k = \frac{1}{y_k^T s_k}. \tag{6.18}$$

For more information on this topic, see e.g. Fletcher [17], Bonnans et al. [6] or Nocedal and Wright [38].

6.3.1 The LBFGS method

When the problem size is large, the BFGS algorithm requires much memory to store the $N_x \times N_x$ large Hessian approximation matrices. This motivates the use of methods that are able to approximate these matrices, using less memory. The limited memory BFGS method (LBFGS method) is a version of the BFGS method for large problems. This method, differs from the BFGS method in the way that the matrix update is performed. Instead of storing the matrix H_k, the m most recent pairs $\{s_i, y_i\}$ is stored at all times. This approach is well functioning for large problems, since studies have shown that small values of m give satisfactory results. In [9] it is claimed that for $m \in [3, 7]$ works well. We give here an outline of the procedure. Firstly we define the correction matrices

$$Y_k = [y_{k-m}, \ldots, y_{k-1}], \tag{6.19}$$

and

$$S_k = [s_{k-m}, \ldots, s_{k-1}]. \tag{6.20}$$

If we let Θ be a positive parameter and we have that $s_i^T y_i > 0$ for $i \in \{k-m, \ldots, k-1\}$, the BFGS matrix is

$$B_k = \Theta I - W_k M_k W_k^T, \tag{6.21}$$

where

$$W_k = [Y_k \quad \Theta S_k], \tag{6.22}$$

$$M_k = \begin{bmatrix} -D_k & L_k^T \\ L_k & \Theta S_k^T S_k \end{bmatrix}^{-1}, \tag{6.23}$$

and where L_k and D_k are $m \times m$ matrices given by

$$(L_k)_{i,j} = \begin{cases} (s_{k-m-1+i})^T (y_{k-m-1+j}) & \text{if } i > j \\ 0 & \text{else} \end{cases} \tag{6.24}$$

and

$$D_k = diag[s_{k-m}^T y_{k-m}, \ldots, s_{k-1}^T y_{k-1}]. \tag{6.25}$$

As explained in [10], since the matrix $M_k \in \mathbb{R}^{2m \times 2m}$, the cost of computing the inverse in (6.23) is negligible in comparison to other costs. In a similar manner, there is also a LBFGS representation of the inverse Hessian approximation H_k, given as

$$H_k = \frac{1}{\Theta} I + \bar{W}_k \bar{M}_k \bar{W}_k^T, \tag{6.26}$$

where

$$\bar{W}_k = \begin{bmatrix} \frac{1}{\Theta} Y_k & S_k \end{bmatrix}, \tag{6.27}$$

$$\bar{M}_k = \begin{bmatrix} 0 & -R_k^{-1} \\ -R_k^{-T} & R_k^{-T} (D_k + \frac{1}{\Theta} Y_k^T Y_k) R_k^{-1} \end{bmatrix}^{-1}, \tag{6.28}$$

and

$$(R_k)_{i,j} = \begin{cases} (s_{k-m-1+i})^T (y_{k-m-1+j}) & \text{if } i \leq j \\ 0 & \text{else} \end{cases} \tag{6.29}$$

To maintain the positive definiteness of the BFGS matrix, we must have that $s_k^T y_k > 0$. However, there is no guarantee that this condition always holds, for the LBFGS method. Consequently, the pair $\{s_k, y_k\}$ is discarded if

$$s_k^T y_k \leq \epsilon ||y||^2, \tag{6.30}$$

for some small constant ϵ.

6.4 Step Length

After having decided on which method to use in order to find a search direction, we need to decide upon how far we shall go along the direction of search. That is, we need to find the step length α_k in (6.3). As previously stated, we find α_k from (6.5), and there are several strategies to solve this problem. In this work we have decided to use the Wolfe conditions, as these conditions have some favourable qualities when used in combination with the BFGS method.

6.4.1 Wolfe conditions

For the step length α_k, we need to impose some conditions on it to insure convergence. Firstly it is important that the step length α_k gives sufficient decrease in the objective function, measured by

$$J(x_k + \alpha_k p_k) \leq J(x_k) + c_1 \alpha_k \nabla J(x_k)^T p_k, \tag{6.31}$$

where $c_1 \in \{0, 1\}$ is a constant. By only imposing the sufficient decrease condition above, we may encounter the problem of having very small step lengths, leading to a slow convergence rate. Thus in order to avoid too small step lengths, we demand that the curvature condition is fulfilled,

$$\nabla J(x_k + \alpha_k p_k)^T p_k \geq c_2 \nabla J(x_k)^T p_k, \tag{6.32}$$

where the constant $c_2 \in \{c_1, 1\}$. This assures that the gradient of the objective function is decreased as well, and these two conditions constitutes the Wolfe conditions.

The Wolfe Conditions

$$J(x_k + \alpha_k p_k) \leq J(x_k) + c_1 \alpha_k \nabla J(x_k)^T p_k, \tag{6.33}$$
$$\nabla J(x_k + \alpha_k p_k)^T p_k \geq c_2 \nabla J(x_k)^T p_k, \tag{6.34}$$

However, it is possible that a step length satisfying the Wolfe conditions, is not close to a local minimiser of the problem, but this can be remedied by changing the curvature condition, giving what is called the strong Wolfe conditions,

The Strong Wolfe Conditions

$$J(x_k + \alpha_k p_k) \leq J(x_k) + c_1 \alpha_k \nabla J(x_k)^T p_k, \tag{6.35}$$
$$|\nabla J(x_k + \alpha_k p_k)^T p_k| \leq c_2 |\nabla J(x_k)^T p_k|, \tag{6.36}$$

When BFGS is used to find the search directions and the step lengths satisfies the Wolfe conditions, the convergence is super-linear see for example [6].

Chapter 7

Constrained Optimisation

Both of the problems considered in this thesis, can be formulated as optimal control problems, which are a special type of constrained optimisation problems. Thus, before we begin to discuss how to solve the optimal control problem, it is useful to discuss constrained optimisation first. Furthermore our optimal control problems are subject to additional constraints and we will here go in to detail on how these are handled.

7.1 Constrained Optimisation Problem

Often it is of interest to minimize a real valued function, $J : \Omega \to \mathbb{R}$ defined on an open set $\Omega \in \mathbb{R}^{N_x}$, subject to a set of constraints on the variables x. We can divide our constraints into two groups of either equality constraints $c_i(x) = 0$, for $i \in E$ or inequality constraints $c_i(x) \leq 0$, for $i \in I$, where we suppose that the index sets E and I have m_E and m_I elements, respectively. The constraints can be written on vectorial form as,

$$c_E(x) = 0, \tag{7.1}$$

$$c_I(x) \leq 0, \tag{7.2}$$

meaning that all elements of $c_E(x) \in \mathbb{R}^{m_E}$ must be zero and that all elements of $c_I(x) \in \mathbb{R}^{m_I}$ must be non-positive. The objective is to find an x that minimises $J(x)$ on the feasible set

$$X = \{x \in \Omega | c_E(x) = 0, c_I \leq 0\}, \tag{7.3}$$

and we can write problem as

$$\min_{x \in \Omega} J(x) \quad \text{subject to} \quad \begin{cases} c_E(x) & = 0 \\ c_I(x) & \leq 0 \end{cases} \tag{7.4}$$

Furthermore we call the inequality constraints which are zero at a point x, for active at x, and we define the set of active constraints as

$$I^0(x) = \{i \in I | c_i(x) = 0\}. \tag{7.5}$$

7.2 First Order Optimality Conditions

In order to define the first order optimality conditions it is convenient to introduce the Lagrangian, L, of the problem (7.4) as

$$L(x, \mu) = J(x) + \mu_I^T c_I(x) + \mu_E^T c_E(x), \tag{7.6}$$

where $\mu = (\mu_I, \mu_E)$ are the Lagrangian multipliers. For our first order optimality conditions to be fulfilled, it is required that our constraints are qualified at the solution. There are several sufficient conditions for qualification, and we list some of them here c.f. [6]

Constraint qualification

We say that the constraints are qualified at x if one of the following conditions is satisfied

$$\text{(A-CQ)} \quad c_{E\cup I^0(x)} \quad \text{is affine in a neighborhood of } x \tag{7.7}$$

$$\text{(S-CQ)} \quad \text{Slater's condition} \tag{7.8}$$

$$\bullet\ c_E \text{ is affine with } c_E'(x) \text{ surjective,} \tag{7.9}$$

$$\bullet \text{ the components of } c_{I^0(x)} \text{ are convex,} \tag{7.10}$$

$$\bullet \text{ there exist a point } \hat{x} \in X \text{ so that } c_{I^0(x)}(\hat{x}) < 0. \tag{7.11}$$

$$\text{(LI-CQ)} \quad \text{The gradients of the active constraints,} \tag{7.12}$$

$$\nabla_x c_i(x), \text{ for } i \in E \cup I^0(x) \text{ are linearly independent.} \tag{7.13}$$

$$\text{(MF-CQ)} \quad \text{Mangasarian-Fromovitz Qualification, if} \tag{7.14}$$

$$\sum_{i\in E\cup I^0(x)} \alpha_i \nabla_x c_i(x) = 0, \text{ with } \alpha_i \geq 0 \text{ for } i \in I^0(x), \tag{7.15}$$

$$\text{then } \alpha_i = 0 \text{ for all } i \in E \cup I^0(x). \tag{7.16}$$

If $x^\star$ is a local solution to (7.4), J and c are Gâteaux differentiable at $x^\star$ and the constraints are qualified, then there exist a Lagrangian multiplier $\mu^\star$ so that the following Karush Kuhn Tucker (KKT) conditions hold,

Karush Kuhn Tucker Conditions

$$\nabla_x L(x^\star, \mu^\star) = 0 \tag{7.17}$$
$$c_E(x^\star) = 0, \tag{7.18}$$
$$c_I(x^\star) \leq 0, \tag{7.19}$$
$$\mu_I^\star \geq 0, \tag{7.20}$$
$$\mu_I^\star c_I(x^\star) = 0. \tag{7.21}$$

These conditions, assures that the solution is a stationary point, but for this to be a minimum we need second order conditions.

7.3 Second Order Optimality Conditions

We state here the second order sufficient conditions. Assume that $\{x^\star, \mu^\star\}$ satisfies the KKT conditions, and that

$$d^T \nabla^2_{xx} L(x^\star, \mu^\star) d > 0, \quad \text{for} \quad d \in C_\star\{0\}, \tag{7.22}$$

then $x^\star$ is a local solution of (7.4). This assures that our solution is local minimum and not a local maximum solution.

7.4 Inequality constraints

There are several different ways of handling inequality constraints, where one can use for example slack variables etc. But, in our problems, the only inequality constraints which are part of our problem formulations are simple bounds on the variables. Bounds are much easier to handle than general inequality constraints, and we will discuss here how these bounds are dealt with.

7.4.1 Bounds on the variables

Bound constrained problems can be written as

$$\min_{x \in \Omega} J(x) \quad \text{subject to} \quad l \leq x \leq u, \tag{7.23}$$

and this is a somewhat simpler problem than the general inequality constrained optimisation problem. In this work we have used a java version of the LBFGS method for bound constrained optimisation, described in detail in [10], [49] and [9], where the bounds on the variables are handled by projected gradients. Below we give a description of this strategy.

7.4.2 Projected Gradients

At iterate k we have available the current estimate x_k, so that $l \leq x_k \leq u$, the function value $J_k = J(x_k)$, the gradient $\nabla J_k = \nabla J(x_k)$ and the BFGS matrix B_k and H_k. Now we want to find a new estimate, x_{k+1}, which lies within the bounds. We are using BFGS, so we form a quadratic model of J at x_k,

$$m_k(x) = J(x_k) + \nabla J_k^T (x - x_k) + \frac{1}{2}(x - x_k)^T B_k (x - x_k). \tag{7.24}$$

Furthermore we define the piecewise linear path

$$x(t) = P(x_k - t\nabla J_k, l, u), \tag{7.25}$$

where P is the projection of $x_k - t\nabla J_k$ onto the feasible region, so that

$$P(x, l, u)_i = \begin{cases} l_i & \text{if} \quad x_i < l_i \\ x_i & \text{if} \quad x_i \in [l_i, u_i] \\ u_i & \text{if} \quad x_i > u_i \end{cases} \tag{7.26}$$

First we calculate the generalized Cauchy point x^c, which is defined as the local minimiser of the piece-wise quadratic

$$q_k(t) = m_k(x(t)). \tag{7.27}$$

The variables which are at one of the bounds at x^c, form what is called the active set, $I^{l,u}(x^c)$. We hold these variables fixed and then solve the following problem for the free variables

$$\min\{m_k(x) : x_i = x_i^c, \ \forall i \in I^{l,u}(x^c)\} \tag{7.28}$$

$$\text{subject to} \quad l_i \leq x_i \leq u_i, \ \forall i \notin I^{l,u}(x^c). \tag{7.29}$$

First, we solve (7.28), ignoring the bounds on the free variables. Then, the path is truncated towards the solution, so that it satisfies the bounds(7.29). When we have found an approximate solution, $\bar{x}_k$ in this way, we compute the new x_{k+1} by performing a line search along the direction $d_k = \bar{x}_{k+1} - x_k$, so that the step length satisfies the strong Wolfe conditions. Finally, we calculate a new gradient $\nabla J(x_{k+1})$, update the BFGS matrices, B_{k+1} and H_{k+1}, and repeat the process.

7.5 Equality constraints

In our formulation of the production optimisation problem, equality constraints on the control variables is part of problem. To handle this we make use of a transformation of variables so that the dimension of the problem is reduced by one and the problem is without these constraints.

7.5.1 The additional Constraints on the Control Variables

The production optimisation problem is formulated with a set of equality constraints on the control variables, equations (4.4) and (4.5), which state the sum of the individual injection rates must equal the total injection rate and that the sum of the individual production rates must equal the total production rate at all times. We restate these constraints here,

$$\sum_{i=1}^{N_{inj}} v_i^n = 1, \quad \text{for} \quad n = 1, \ldots, N, \tag{7.30}$$

and

$$\sum_{i=1+N_{inj}}^{N_{prod}+N_{inj}} v_i^n = 1, \quad \text{for} \quad n = 1, \ldots, N. \tag{7.31}$$

In addition, we also have inequality constraints given in (4.6),

$$0 \leq v_i^n \leq 1, \quad \text{for} \quad i = 1, \ldots, N_{inj} + N_{prod}, \tag{7.32}$$

for $n = 1 \ldots N$. To incorporate these equality constraints, we define a set of $(N_{inj} + N_{prod} - 2) \cdot N$ functions

$$0 \leq \xi_1^n, \ldots, \xi_{N_{inj}+N_{prod}-2}^n \leq 1, \quad \text{for} \quad n = 1 \ldots N, \tag{7.33}$$

and denote $\xi^n = \{\xi_i^n\}_{i=1}^{N_{inj}+N_{prod}-2}$. Now the v^n are given in terms of ξ^n by the relations,

$$\begin{aligned}
v_1^n &= \xi_1^n \\
v_2^n &= (1-\xi_1^n)\xi_2^n \\
v_3^n &= (1-\xi_1^n)(1-\xi_2^n)\xi_3^n \\
&\vdots \\
v_{N_{inj}-1}^n &= \xi_{N_{inj}-1}^n \prod_{i=1}^{N_{inj}-2} (1-\xi_i^n) \\
v_{N_{inj}}^n &= \prod_{i=1}^{N_{inj}-1} (1-\xi_i^n),
\end{aligned} \tag{7.34}$$

and

$$\begin{aligned} v^n_{1+N_{inj}} &= \xi^n_{N_{inj}} \\ v^n_{2+N_{inj}} &= (1-\xi^n_{N_{inj}})\xi^n_{1+N_{inj}} \\ v^n_{3+N_{inj}} &= (1-\xi^n_{N_{inj}})(1-\xi^n_{1+N_{inj}})\xi^n_{2+N_{inj}} \\ &\vdots \\ v^n_{N_{prod}+N_{inj}-1} &= \xi^n_{N_{prod}+N_{inj}-2} \prod_{i=1}^{N_{prod}+N_{inj}-3} (1-\xi^n_i) \\ v^n_{N_{prod}+N_{inj}} &= \prod_{i=1}^{N_{prod}+N_{inj}-2} (1-\xi^n_i). \end{aligned} \tag{7.35}$$

Now we observe that, when defining the rates as in (7.34) and (7.35) , (7.30) and (7.31) hold for all

$$0 \le \xi^n_1, \dots, \xi^n_{N_{inj}+N_{prod}-2} \le 1. \tag{7.36}$$

Using ξ^n instead of v^n, we have a optimisation problem without the linear equality constraints. However the inequality constraints in (7.33) are transformed to the inequality constraints in (7.36), which we handle with a projected method, described in the previous section. To solve the optimal control problems, we need to calculate the derivative of the augmented Lagrangian with respect to the controls. The derivative of the augmented Lagrangian with respect to v has been described in chapter 3, but since we have changed the variables from v to ξ we must find their derivatives also. To find the derivative, we have by the chain rule that for $n = 1, \dots, N$,

$$\frac{\partial L^n_c}{\partial \xi^n_i} = \sum_{j=1}^{N_{inj}} \frac{\partial L^n_c}{\partial v^n_j} \frac{\partial v^n_j}{\partial \xi^n_i}, \quad \text{for} \quad 0 \le i \le N_{inj} - 1, \tag{7.37}$$

and

$$\frac{\partial L^n_c}{\partial \xi^n_i} = \sum_{j=1+N_{inj}}^{N_{inj}+N_{prod}} \frac{\partial L^n_c}{\partial v^n_j} \frac{\partial v^n_j}{\partial \xi^n_i}, \quad \text{for} \quad N_{inj} \le i \le N_{inj} + N_{prod} - 2. \tag{7.38}$$

The derivative $\frac{\partial L^n_c}{\partial v^n_j}$, we already know, see paper B, and the other derivative is given by

$$\frac{\partial v^n_j}{\partial \xi^n_i} = \begin{cases} 0 & \text{if } i > j \\ \prod_{k=1}^{j-1}(1-\xi^n_k) & \text{if } i = j, \text{ and } j, i < N_{inj} \\ -\frac{\xi^n_j}{1-\xi^n_i}\prod_{k=1}^{j-1}(1-\xi^n_k) & \text{if } i < j, \text{ and } j, i < N_{inj} \\ -\frac{1}{1-\xi^n_i}\prod_{k=1}^{j-1}(1-\xi^n_k) & \text{if } i < j = N_{inj} \end{cases}$$

This method of handling the equality constraints (7.30) and (7.31) is used in all the paper contained in part II of the thesis.

Chapter 8

Optimal Control

Optimal control problems is the problem of minimising a functional, $J(x)$, with respect to $x \in \mathbb{R}^{N_x}$, where there are a set of equality constraints $e(x) = 0$, so that $e(x) \in \mathbb{R}^{M_s}$, where $M_s < N_x$. It is then possible to partition our variables, x, into a set of control variables $v \in \mathbb{R}^{N_x - M_s}$ and a set of state variables $u \in \mathbb{R}^{M_s}$, so that $x = \{v, u\}$. The constraints $e(x) = 0$ are called the state equations. These type of problems appear in many important engineering situations. And it is possible, as shown earlier, to regard both the production optimisation problem and the history matching problem as optimal control problems. For both these problems we want to optimise a functional J, under the constraints that the reservoir flow equations are zero. The state variables are the pressures and saturations at each grid cell of the reservoir, and the control variables are the injection/production rates or the permeabilities for the production optimisation problem or the history matching problem, respectively.

8.1 Problem Formulation

Let us define a set of control variables $v^n = \{v_1^n, \ldots, v_{M_c}^n\} \in \mathbb{R}^{M_c}$, and a set of state variables $u^n = \{u_1^n, \ldots, u_{M_s}^n\} \in \mathbb{R}^{M_s}$, for $n = 1, \ldots, N$, where n is the discrete time step, N is the total number of time steps as before, $M_c = N_x - M_s$ is the number of control variables at time step n and M_s is the number of state variables at time step n. We want to solve

$$\min_{v,u} J(v,u) \qquad \text{subject to} \qquad e^n(v^n, u^n, u^{n-1}) = 0, \text{ for } n = 1, \ldots, N, \tag{8.1}$$

where $e^n \in \mathbb{R}^{M_s}$ are nonlinear functions in u^n and v^n, and the functional J is given by

$$J = \sum_{n=1}^{N} J^n(v^n, u^n). \tag{8.2}$$

Since both $e^n \in \mathbb{R}^{M_s}$ and $u^n \in \mathbb{R}^{M_s}$, the e^n are defining equations for the state variables, i.e. if we have a set of control variables, it is possible to find the state variables by solving the state equations, $e^n = 0$, for $n = 1, \ldots, N$. Optimal control problems are a sort of constrained optimisation problems, and we can therefore state the Lagrangian of the problem,

$$L = \sum_{n=1}^{N} L^n, \tag{8.3}$$

where

$$L^n = J^n + \mu^{nT} e^n(v^n, u^n, u^{n-1}). \tag{8.4}$$

Furthermore, we define the augmented Lagrangian by

$$L_c = \sum_{n=1}^{N} L_c^n, \tag{8.5}$$

where

$$L_c^n = J^n + (\mu^n + \frac{c}{2} e^n(v^n, u^n, u^{n-1}))^T e^n(v^n, u^n, u^{n-1}), \tag{8.6}$$

and c is a positive scalar, so that $c > 0$. Since our problem is a constrained optimisation problem, the KKT conditions hold at the solution of (8.1), and we restate them here for the optimal control problem.

Karush Kuhn Tucker conditions Suppose that $\{v^\star, u^\star\}$ is a solution of (8.1). Then there exists a set of vectors $\mu^\star$ such that the following conditions are fulfilled at the point $\{v^\star, u^\star\}$, for all n

$$\begin{aligned} \nabla_{u^n} L(v^\star, u^\star, \mu^\star) &= 0, & (8.7)\\ \nabla_{v^n} L(v^\star, u^\star, \mu^\star) &= 0, & (8.8)\\ \nabla_{\mu^n} L(v^\star, u^\star, \mu^\star) &= 0. & (8.9) \end{aligned}$$

Since equation (8.9) states that the state equations are zero, the KKT conditions must also be valid for the augmented Lagrangian, so that these conditions are equivalent to

$$\begin{aligned} \nabla_{u^n} L_c(v^\star, u^\star, \mu^\star) &= 0, & (8.10)\\ \nabla_{v^n} L_c(v^\star, u^\star, \mu^\star) &= 0,\\ \nabla_{\mu^n} L_c(v^\star, u^\star, \mu^\star) &= 0. \end{aligned}$$

In order to find the solution of (8.1), we need to solve the KKT system (8.7) or equivalently (8.10). But it must be noticed, however, that the KKT conditions state that the solution of our problem is a stationary point of the Lagrangian and of the augmented Lagrangian, but they do not tell us if we should search for a maximum of a minimum. In fact, the solution is a saddle point of the Lagrangian and the augmented Lagrangian, which we show next.

8.2 The Saddle Points of L and L_c

Theorem Let $\{v^\star, u^\star, \mu^\star\}$ be a saddle point of L. Then $\{v^\star, u^\star, \mu^\star\}$ is solution of (8.1). Furthermore $\{v^\star, u^\star, \mu^\star\}$ is also a saddle point of L_c, for all $c > 0$. If $\{v^\star, u^\star, \mu^\star\}$ is a saddle point of L_c, then $\{v^\star, u^\star, \mu^\star\}$ is a solution of (8.1).

Proof:

1. Assume that $\{v^\star, u^\star, \mu^\star\}$ is a saddle point of L. Then we have that

$$\begin{aligned} L(v^\star, u^\star, \gamma) &\leq L(v^\star, u^\star, \mu^\star) \leq L(p, q, \mu^\star), \\ \forall\{p, q, \gamma\} &\in V \times H \times H, \quad \{v^\star, u^\star, \mu^\star\} \in V \times H \times H. \end{aligned} \tag{8.11}$$

From the first inequality (8.11) and from (8.4) it follows that

$$\sum_{n=1}^{N} \gamma^{nT} e^n(v^{\star n}, u^{\star n}, u^{\star n-1}) \leq \sum_{n=1}^{N} \mu^{\star nT} e^n(v^{\star n}, u^{\star n}, u^{\star n-1}), \quad \forall \mu \in H. \tag{8.12}$$

This implies that

$$e^n(v^{\star n}, u^{\star n}, u^{\star n-1}) = 0, \quad \text{for} \quad n = 1, \ldots N. \tag{8.13}$$

From the second inequality of (8.11), we have that

$$L(v^\star, u^\star, \mu^\star) \leq L(p, q, \mu^\star), \quad \forall\{p, q\} \in V \times H. \tag{8.14}$$

Using equation (8.13), this is equivalent to

$$J(v^\star, u^\star) \leq L(p, q, \mu^\star), \quad \forall\{p, q\} \in V \times H. \tag{8.15}$$

Let us choose $\{p, q\}$ so that $e^n(p^n, q^n) = 0$ for all $n = 1, \ldots, N$ in (8.15). It then follows that

$$\begin{aligned} J(v^\star, u^\star) &\leq L(p, q, \mu^\star) = J(p, q), \\ \forall\{p, q\} &\in \{\{p, q\} \in V \times H \mid e^n(p^n, q^n) = 0, \forall n\}. \end{aligned} \tag{8.16}$$

And we see that $\{v^\star, u^\star\}$ is a solution to the optimisation problem (8.1).

2. Since $e^n(v^{\star n}, u^{\star n}, u^{\star n-1}) = 0, \forall n$, we have that

$$L_c(v^\star, u^\star, \gamma) = L(v^\star, u^\star, \gamma) = J(v^\star, u^\star), \quad \forall \gamma \in H. \tag{8.17}$$

From equation (8.17) and the second inequality of (8.11) it then follows that

$$L_c(v^\star, u^\star, \gamma) = L_c(v^\star, u^\star, \mu^\star) \leq L(p, q, \mu^\star), \quad \forall \{p, q, \gamma\} \in V \times H \times H. \tag{8.18}$$

Since $L_c(p, q, \mu^\star) = L(p, q, \mu^\star) + \sum_{n=1}^{N} \frac{c}{2} e^{nT} e^n$, we get

$$L_c(v^\star, u^\star, \gamma) \leq L_c(v^\star, u^\star, \mu^\star) \leq L_c(p, q, \mu^\star), \quad \forall \{p, q, \gamma\} \in V \times H \times H, \tag{8.19}$$

which proves that $\{v^\star, u^\star, \gamma\}$ is also a saddle point of L_c on $V \times H \times H$.

3. Assume that $\{v^\star, u^\star, \mu^\star\}$ is a saddle point of L_c, with $c > 0$. Then we have that

$$\begin{aligned} L_c(v^\star, u^\star, \gamma) &\leq L_c(v^\star, u^\star, \mu^\star) \leq L_c(p, q, \mu^\star), \\ \forall \{p, q, \gamma\} &\in V \times H \times H, \quad \{v^\star, u^\star, \mu^\star\} \in V \times H \times H. \end{aligned} \tag{8.20}$$

The first inequality of (8.20) gives that

$$\sum_{n=1}^{N} \gamma^{nT} e^n(v^{\star n}, u^{\star n}, u^{\star n-1}) \leq \sum_{n=1}^{N} \mu^{\star nT} e^n(v^{\star n}, u^{\star n}, u^{\star n-1}), \quad \forall \mu \in H, \tag{8.21}$$

which implies that

$$e^n(v^{\star n}, u^{\star n}, u^{\star n-1}) = 0, \quad \text{for} \quad n = 1, \ldots N. \tag{8.22}$$

From the second inequality (8.20) and from (8.22), it follows that

$$J(v^\star, u^\star) = L_c(v^\star, u^\star, \mu^\star) \leq L_c(p, q, \mu^\star), \quad \forall \{p, q\} \in V \times H. \tag{8.23}$$

Let us denote the solution of (8.1) as $\{\tilde{v}, \tilde{u}\} \in V \times H$. Since $\{\tilde{v}, \tilde{u}\}$ is the solution of (8.1), we have that

$$L_c(\tilde{v}, \tilde{u}, \mu^\star) = J(\tilde{v}, \tilde{u}) \leq J(p, q), \tag{8.24}$$

$$\forall \{p, q\} \in \{\{p, q\} \in V \times H \mid e^n(p^n, q^n) = 0, \forall n\}. \tag{8.25}$$

Since this holds for all $\{p, q\} \in \{\{p, q\} \in V \times H \mid e^n(p^n, q^n) = 0, \forall n\}$, it also holds for $\{v^\star, u^\star\}$, so that

$$L_c(\tilde{v}, \tilde{u}, \mu^\star) = J(\tilde{v}, \tilde{u}) \leq J(v^\star, u^\star). \tag{8.26}$$

Thus, it follows from (8.23) and (8.26) that

$$J(v^\star, u^\star) \leq J(\tilde{v}, \tilde{u}) \leq J(v^\star, u^\star), \tag{8.27}$$

such that $\{v^\star, u^\star\} = \{\tilde{v}, \tilde{u}\}$, and hence $\{v^\star, u^\star\}$ is a solution of (8.1).

8.3 Solving the Optimal Control Problem

All of the methods presented further on in this book, on how to solve the optimal control problem are based on searching for the saddle point of either the Lagrangian or the augmented Lagrangian. We will here try to give a survey of the different algorihtms that solve the optimal control problem. The algorithm called the KKT algorithm has not been used before, to our knowledge, and it is our main contribution. It is used in papers A,C and D. Furthermore we have developed a new marching scheme of the augmented Lagrangian, which is the algorithm called New Marching Scheme. It is an improvement of earlier marching schemes, and is used in B.

These methods can be regarded as stemming from one common general approach, which result in a variety of methods which we describe here. It is possible to regard the saddle point problem as a constrained problem, in one of the following ways,

$$\min_{v} L_c(v, \hat{u}, \hat{\mu}) \quad \text{subject to} \quad \begin{cases} \hat{\mu} = \operatorname{argmax}_{\mu} L_c(v, u, \mu) \\ \hat{u} = \operatorname{argmin}_{u} L_c(v, u, \mu) \end{cases} \tag{8.28}$$

$$\min_{u} L_c(\hat{v}, u, \hat{\mu}) \quad \text{subject to} \quad \begin{cases} \hat{\mu} = \operatorname{argmax}_{\mu} L_c(v, u, \mu) \\ \hat{v} = \operatorname{argmin}_{v} L_c(v, u, \mu) \end{cases} \tag{8.29}$$

or

$$\max_{\mu} L_c(\hat{v}, \hat{u}, \mu) \quad \text{subject to} \quad \begin{cases} \hat{v} = \operatorname{argmin}_{v} L_c(v, u, \mu) \\ \hat{u} = \operatorname{argmin}_{u} L_c(v, u, \mu) \end{cases} \tag{8.30}$$

The common idea is to maximise or minimise the augmented Lagrangian subject to that the optimisations in the two other variables are fulfilled. This gives us three possibilities, but since the state variables and the Lagrangian multipliers are in the same space, these may be interchanged, so that, for example, one finds μ so that it minimises L_c with respect to u. As we show later, this gives a total of four possibilities. The details are given in this section as we proceed, and we start with the familiar adjoint method.

8.4 The Adjoint Method

We use the adjoint method as a method for comparison of the results obtained with the methods are testing in the papers in part II of this dissertation. In the adjoint method one regards the saddle point problem as the problem

$$\min_{v} L_c(v, \hat{u}, \hat{\mu}) \quad \text{subject to} \quad \begin{cases} \hat{\mu} = \operatorname{argmax}_{\mu} L_c(v, u, \mu) \\ \hat{u} = \operatorname{argmin}_{u} L_c(v, u, \mu) \end{cases} \tag{8.31}$$

Here the first of the constraints simply states that the state equations, $e^n(v,u) = 0$ for $n = 1,\ldots,N$, are fulfilled, so that this can be rewritten as

$$\min_v L_c(v,\hat{u}) \quad \text{subject to} \quad \begin{cases} e^n(v,u) = 0, \text{ for } n = 1,\ldots,N \\ \hat{u} = \text{argmin}_u L_c(v,u) \end{cases} \tag{8.32}$$

Since one of the constraints is that the state equations are fulfilled, $e^n(v,u) = 0$ for $n = 1,\ldots,N$, the augmented Lagrangian reduces to the Lagrangian. It is possible to find $\hat{u}$ so that $e^n(v,\hat{u}) = 0$, for $n = 1,\ldots,N$ and μ so that $\hat{u} = \text{argmin}_u L_c(v,u)$, but since $e^n(v,u)$ is independent of μ, this is the only way of doing it, i.e. we cannot find μ so that $e^n(v,u) = 0$, for $n = 1,\ldots,N$. Now to classify this strategy, we consider the following. The conditions for a solution of the optimal control variables, includes the control variables, $v \in V$, the state variables, $u \in U$, and the Lagrangian multipliers, $\mu \in U$. But if we let the control variables be the only independent variables so that $u = u(v)$ and $\mu = \mu(v)$, then we have that

$$\frac{dL}{dv} = \frac{\partial L}{\partial v} + \frac{\partial L}{\partial u}\frac{du}{dv} + \frac{\partial L}{\partial \mu}\frac{d\mu}{dv}. \tag{8.33}$$

If we can find u such that

$$\frac{\partial L}{\partial \mu^n} = e^n(v^n,u^n,u^{n-1}) = 0, \quad \text{for} \quad \text{n} = 1,\ldots,\text{N}, \tag{8.34}$$

and μ so that

$$\begin{aligned} \frac{\partial L}{\partial u^n} &= \frac{\partial J^n}{\partial u^n}(v^n,u^n) + \frac{\partial e^n}{\partial u^n}^T(v^n,u^n)\mu^n + \frac{\partial e^{n+1}}{\partial u^n}^T \mu^{n+1} = 0, \text{ for n} = 1,\ldots,\text{N}-1 \\ \frac{\partial L}{\partial u^N} &= \frac{\partial J^N}{\partial u^N}(v^N,u^N) + \frac{\partial e^N}{\partial u^N}^T(v^N,u^N)\mu^N = 0 \end{aligned} \tag{8.35}$$

then we will have that

$$\frac{dL}{dv} = \frac{\partial L}{\partial v}. \tag{8.36}$$

Furthermore, by using equation (8.34), we have that

$$L(v,u) = J(v,u), \tag{8.37}$$

and thus it follows that

$$\frac{dJ}{dv} = \frac{dL}{dv} = \frac{\partial L}{\partial v}. \tag{8.38}$$

Solving the problem in (8.34) is the same as solving the forward problem, and the problems in equations (8.35) is a linear inverse problem, that is fairly easy to solve. The gradient in (8.38) can then be used in a gradient based minimisation procedure, using for example the LBFGS or some other method for optimisation, see for example Nocedal and Wright [38] or Bonnans et al. [6]. The algorithm is as follows,

Algorithm 1 The Adjoint Method

1. Choose $v_0 = \{v_0^n\}_{n=1}^N$. For $k = 1, 2, \ldots$ do:
2. For $n = 1, 2, \ldots, N$, find u_k^n such that
$$e^n(u_k^n, u_k^{n-1}, v_{k-1}^n) = 0$$
3. Find λ_k^N such that
$$-\frac{\partial \boldsymbol{J}^N}{\partial u^N} + \lambda_k^{N\,T} \frac{\partial e^N}{\partial u^N} = 0,$$
4. For $n = N-1, N-2, \ldots, 1$, find λ_k^n, such that
$$-\frac{\partial \boldsymbol{J}^n}{\partial u^n} + \lambda_k^{n\,T} \frac{\partial e^n}{\partial u^n} + \lambda_k^{n+1\,T} \frac{\partial e^{n+1}}{\partial u^n} = 0,$$
5. Finally we find the gradient of $-\boldsymbol{J}$ as
$$-\frac{d\boldsymbol{J}^n}{dv^n} = \frac{\partial L_c}{\partial v^n} = -\frac{\partial \boldsymbol{J}^n}{\partial v^n} + \lambda_k^{n\,T} \frac{\partial e^n}{\partial v^n} = 0,$$
and use this gradient in a gradient-based minimisation procedure to find $v_k = \{v_k^n\}_{n=1}^N$.

The adjoint method has been used to solve the History matching problem, several times, see e.g. , and it has also been used to solve the production optimisation problem, see for example [7] and [33].

8.5 Augmented Lagrangian Methods

The augmented Lagrangian method, or the method of multipliers was first introduced independently by Hestenes in [24] and by Powell in [42], where the constrained minimisation problem is solved as a sequence of minimization problems. For the augmented Lagrangian methods we regard the saddle point problem as a maximum problem, where the minimum problems are treated as constraints, so that we get the second alternative listed in initially

$$\max_{\mu} L_c(\hat{v}, \hat{u}, \mu) \quad \text{subject to} \quad \begin{cases} \hat{v} = \operatorname{argmin}_v L_c(v, u, \mu) \\ \hat{u} = \operatorname{argmin}_u L_c(v, u, \mu) \end{cases} \tag{8.39}$$

In standard augmented Lagrangian methods, one finds μ from the outer maximisation problem and u from the constraint that u must minimise the augmented

Lagrangian functional. As for the adjoint method, we can let two of the variables be dependent of the third, so that $v = v(\mu)$ and $u = u(\mu)$. Then we find that

$$\frac{dL}{d\mu} = \frac{\partial L}{\partial \mu} + \frac{\partial L}{\partial u}\frac{du}{d\mu} + \frac{\partial L}{\partial v}\frac{dv}{d\mu}. \tag{8.40}$$

If we find u and v so that

$$\frac{\partial L}{\partial u} = 0 \tag{8.41}$$

$$\frac{\partial L}{\partial v} = 0, \tag{8.42}$$

then it follows that

$$\frac{dL}{d\mu} = \frac{\partial L}{\partial \mu}. \tag{8.43}$$

From the expression of the augmented Lagrangian we have that

$$\frac{\partial L_c}{\partial \mu^n} = e^n(v^n, u^n, u^{n-1}). \tag{8.44}$$

And if we now use the steepest ascend method to solve (8.39), we find the standard augmented Lagrangian method,

Algorithm 2 Augmented Lagrangian Method

1. Choose λ_0, v_0 and $c > 0$
2. For $k = 1, \ldots,$ do:
3. Find u_k and v_k such that

$$u_k = \underset{u}{\operatorname{argmin}}\, L_c(v_{k-1}, u, \lambda_{k-1}) \tag{8.45}$$

and

$$v_k = \underset{v}{\operatorname{argmin}}\, L_c(v, u_k, \lambda_{k-1}) \tag{8.46}$$

4. Update

$$\lambda_k = \lambda_{k-1} + \alpha_c \cdot e(v_k, u_k). \tag{8.47}$$

Here we see that the update in equation (8.47) is a steepest ascend search direction where α_c is the step length. A natural question to ask, is why we need $c > 0$. The reason for this is that, in order for this to work, we need that there exists a local solution of (8.45) and (8.46). By having c larger then a certain value, we can assure that the second derivative of the augmented Lagrangian is positive definite,

so that the second order conditions (7.22) are fulfilled. For the choice of the scalar α_c, it is common practice to set $\alpha_c = c$. However, Bertsekas in [5], has tried to find an optimal step length and it is shown in [4], that the convergence rate may be improved by using a more optimal step length then the choice $\alpha_c = c$. Instead of the steepest ascend method, it is possible to use second order methods to find the search direction. We have not considered this in our work, and we refer the interested reader to for example [5].

8.6 The Augmented Lagrangian Method

In the Augmented Lagrangian method, described in the last section, we need to find v and u simultaneously in equation (8.45) and (8.46). Due to a large number of variables, this may become computationally expensive. One, may then use a Gauss-Seidel strategy and split this minimisation up in two, one for the controls and one for the state variables, and solve them sequentially. In this way, they will converge in the end. The algorithm, goes as follows, using the step length $\alpha_c = c$

Algorithm 3 Uzawa Augmented Lagrangian Method

1. Choose λ_0, v_0 and $c > 0$
2. For $k = 1, \ldots,$ do:
3. Find u_k such that

$$u_k = \underset{u}{\operatorname{argmin}} L_c(v_{k-1}, u, \lambda_{k-1}) \tag{8.48}$$

4. Find v_k such that

$$v_k = \underset{v}{\operatorname{argmin}} L_c(v, u_k, \lambda_{k-1}) \tag{8.49}$$

5. Update

$$\lambda_k = \lambda_{k-1} + c \cdot e(v_k, u_k). \tag{8.50}$$

8.7 Marching Schemes

The global augmented Lagrangian method, has for optimal control problems, the disadvantage that the minimisation in the state variables in (8.48) may be very time consuming, since the number of state variables is usually large. To remedy this, it was proposed in Nilssen et al. [36], [37], [35] to do the minimisation in

(8.48) and in (8.49) sequentially, time step by time step, thereby reducing the number of variables in each minimisation significantly. The idea is that, in stead of doing the minimisation in step (8.48), one should use the approximation, for $n = 1, \ldots, N$,

$$u_k^n = \underset{u^n}{\operatorname{argmin}} L_c^n(v_{k-1}^n, u^n, u_k^{n-1}, \lambda_{k-1}^n), \tag{8.51}$$

and in stead of equation (8.49) one should do for $n = 1, \ldots, N$,

$$v_k^n = \underset{v^n}{\operatorname{argmin}} L_c^n(v^n, u_k^n, u_k^{n-1}, \lambda_{k-1}^n). \tag{8.52}$$

With these modifications to the global algorithm, the resulting algorithm is

Algorithm 4 The Old Marching Scheme

1. Choose v_0, λ_0 and $c > 0$
2. For $k = 1, 2, \ldots$ do:
3. For $n = 1, \ldots, N$, find u_k^n so that

$$u_k^n = \underset{u^n}{\operatorname{argmin}} L_c^n(v_{k-1}^n, u^n, u_k^{n-1}, \lambda_{k-1}) \tag{8.53}$$

4. For $n = 1, \ldots, N$, find v_k^n so that

$$u_k^n = \underset{v^n}{\operatorname{argmin}} L_c^n(v^n, u_k^n, u_k^{n-1}, \lambda_{k-1}) \tag{8.54}$$

5. Update

$$\lambda_k = \lambda_{k-1} + c \cdot e(v_k, u_k). \tag{8.55}$$

In Nilssen et al. [36] and [37], this has been applied successfully, for solving least-squares minimisation problems with constraints. Here we attempt to analyse certain aspects of the old marching scheme, in order to understand it better. Contrary to the global optimisation algorithm witch a solution satisfying the equations (8.10), the old marching scheme finds a solution $\{v_\dagger, u_\dagger, \lambda_\dagger\}$ satisfying the following equations, for $n = 1, \ldots, N$,

$$\begin{aligned} -\frac{\partial J^n}{\partial v^n}(v_\dagger, u_\dagger) + \lambda_\dagger^{nT} \frac{\partial e^n}{\partial v^n}(v_\dagger, u_\dagger) &= 0, \\ -\frac{\partial J^n}{\partial u^n}(v_\dagger, u_\dagger) + \lambda_\dagger^{nT} \frac{\partial e^n}{\partial u^n}(v_\dagger, u_\dagger) &= 0, \\ e^n(v_\dagger, u_\dagger) &= 0. \end{aligned} \tag{8.56}$$

If this is a solution of (8.1), the solution must satisfy the equations (8.10). Thus a solution found by the old marching scheme can only be a solution of (8.1) if

$$\frac{\partial L_c^m}{\partial u^n}(v_\star, u_\star, \lambda_\star) = 0, \quad \text{for} \quad \mathrm{m} \neq \mathrm{n} \quad \text{and} \quad \mathrm{n}, \mathrm{m} \in \{1, \ldots, \mathrm{N}\}, \tag{8.57}$$

and if

$$\frac{\partial L_c^m}{\partial v^n}(v_\star, u_\star, \lambda_\star) = 0, \quad \text{for} \quad \mathrm{m} \neq \mathrm{n} \quad \text{and} \quad \mathrm{n}, \mathrm{m} \in \{1, \ldots, \mathrm{N}\}. \tag{8.58}$$

For both our production optimisation problem and for our history matching problem, (8.58) fulfilled, but (8.57) is only fulfilled for the history matching problem. From our model we have that, for $n = 1, \ldots, N-1$

$$\frac{\partial L_c}{\partial u^n} = \frac{\partial L_c^n}{\partial u^n} + \frac{\partial L_c^{n+1}}{\partial u^n} \tag{8.59}$$

$$= \frac{\partial J^n}{\partial u^n} + \frac{\partial e^n}{\partial u^n}^T (\lambda^n + c \cdot e^n) + \frac{\partial e^{n+1}}{\partial u^n}^T (\lambda^{n+1} + c \cdot e^{n+1}), \tag{8.60}$$

and that

$$\frac{\partial L_c}{\partial u^N} = \frac{\partial L_c^N}{\partial u^N} = \frac{\partial J^N}{\partial u^N} + \frac{\partial e^N}{\partial u^N}^T (\lambda^N + c \cdot e^N). \tag{8.61}$$

Now, for $n = 1, \ldots, N$, if $\frac{\partial J^n}{\partial u^n}(v_\star^n, u_\star^n) = 0$, we see that must have that $\lambda_\star^n = 0$ since $e^n(v_\star^n, u_\star^n, u_\star^{n-1})$, showing that the old marching scheme finds a solution to the problem if we have that

$$\frac{\partial J^n}{\partial u^n}(v_\star^n, u_\star^n) = 0, \quad \text{for} \quad n = 1, \ldots, N. \tag{8.62}$$

Condition (8.62) will typically be fulfilled if we have a least-squares objective function where one expects that the objective function will be close to zero at the minimum. For the problems that was solved by Nilssen et al. in [36] etc. this has been the case. With regards to the history matching problem, we have a least-squares objective function, and if the noise in the data is low, condition (8.62) is approximately true. However, for the production optimisation problem, condition (8.62) does not hold, since the objective function is not a least squares functional. As a result, the old marching scheme will not produce the right answer for the production optimisation problem. As an attempt to improve the old marching scheme, so that it can be used to solved a larger class of problems, we propose in paper B a new marching scheme. Where we have compared it to the old marching scheme.

8.7.1 The new marching scheme

Although the global Uzawa scheme is convergent, it has the drawback of being quite time consuming, since the minimisation problem (8.48) involves a very large number of variables. To remedy this, the old marching scheme splits this minimisation problem into N, one for each time step, smaller minimisation problems in (8.51). Unfortunately the resulting algorithm is only capable to solve a limited number of optimisation problems, and the new marching scheme tries to improve this. In the old marching scheme the optimisation is done sequentially by neglecting the gradient term

$$\frac{\partial L_c^{n+1}}{\partial u^n} = \left(\lambda^{n+1} + c \cdot e^{n+1}\right)^T \frac{\partial e^{n+1}}{\partial u^n}, \tag{8.63}$$

to improve the speed of the global optimisation algorithm. This may work, when condition (8.57) is fulfilled, but since this is not the case for our problem we cannot use the old marching scheme.

Here we present a different way to approximate the minimisation with respect to u in the global optimisation algorithm, such that this sub-minimisation problem can be solved sequentially. We want to do the minimisation with respect to u sequentially, time step by time step, to improve the speed of the algorithm. Since e^{n+1} depends on u^{n+1}, which is unknown at iteration n, we must do some sort of approximation. When this minimisation is done in the old marching scheme, we find u^n such that

$$-\frac{\partial J^n}{\partial u^n} + \left(\lambda^n + c \cdot e^n\right)^T \frac{\partial e^n}{\partial u^n} = 0.$$

But it is not necessary to neglect the term (8.63) entirely, because it is only e^{n+1} which depends on u^{n+1}. We propose instead to find u^n sequentially such that

$$-\frac{\partial J^n}{\partial u^n} + \left(\lambda^n + c \cdot e^n\right)^T \frac{\partial e^n}{\partial u^n} + {\lambda^{n+1}}^T \frac{\partial e^{n+1}}{\partial u^n} = 0. \tag{8.64}$$

In paper B we show that if we find u^n so that

$$u^n = \underset{u^n}{\operatorname{argmin}} \left[-J^n + \left(\lambda^n + \frac{c}{2} e^n\right)^T e^n + {\lambda^{n+1}}^T \tilde{S}^{n+1} \right], \tag{8.65}$$

where $\tilde{S}^{n+1}$ is given by

$$\tilde{S}^{n+1} = [\phi_1 s_1^n, \ldots, \phi_m s_m^n, -\phi_1 s_1^n, \ldots, -\phi_m s_m^n]^T, \tag{8.66}$$

is equivalent to finding u^n such that (8.64) is satisfied, since $\frac{\partial \tilde{S}^{n+1}}{\partial u^n} = \frac{\partial e^{n+1}}{\partial u^n}$, cf. paper B. In fact, if e^n approaches 0 for $n = 1, \ldots, N$ at convergence, this algorithm

will satisfy the KKT conditions of (8.1) and thus solve the problem. Let us define $\tilde{L_c^n}$ by

$$\tilde{L_c^n} = -J^n + \left(\lambda^n + \frac{c}{2}e^n\right)^T e^n + {\lambda^{n+1}}^T \tilde{S}^{n+1}. \tag{8.67}$$

Now the new marching algorithm, which is used in paper B, is as follows.

Algorithm 5 New Marching Scheme

1. Choose $v_0 = \{v_0^n\}_{n=1}^N$, $\lambda_0 = \{\lambda_0^n\}_{n=1}^N$, $c > 0$. For $k = 1, 2, \ldots$ do:
2. For $n = 1, \ldots, N$, find u_k^n such that

$$u_k^n = \underset{u^n}{\operatorname{argmin}}\, \tilde{L_c^n}(v_{k-1}^n, u^n, u_k^{n-1}, \lambda_{k-1}^{n+1}, \lambda_{k-1}^n). \tag{8.68}$$

3. Then, for $n = 1, \ldots, N$ find v_k^n such that

$$v_k^n = \underset{v^n}{\operatorname{argmin}}\, L_c^n(v^n, u_k^n, u_k^{n-1}, \lambda_{k-1}^n), \tag{8.69}$$

4. And finally update the Lagrangian multiplier by

$$\lambda_k^n = \lambda_{k-1}^n + c\, e^n(v_k^n, u_k^n, u_k^{n-1}). \tag{8.70}$$

This will be done iteratively until convergence. And as in the previous optimisation algorithms, equation (8.70) of the algorithm will assure that e^n, for $n = 1, \ldots, N$, tends to zero.

8.8 The KKT method

The KKT method has not to our knowledge been documented before, and the development and testing of this algorithm is the main contribution in this thesis. We have studied it for both the production optimisation problem and the history matching problem in papers A, C and D. In the KKT method we regard the saddle point problem as the problem

$$\max_{\mu} L_c(\hat{v}, \hat{u}, \mu) \quad \text{subject to} \quad \begin{cases} \hat{v} = \operatorname{argmin}_v L_c(v, u, \mu) \\ \hat{u} = \operatorname{argmin}_u L_c(v, u, \mu) \end{cases} \tag{8.71}$$

which is the same as for the augmented Lagrangian method. However, in the augmented Lagrangian method, we find u and v so that

$$\hat{u} = \underset{u}{\operatorname{argmin}}\, L_c(v, u, \mu), \tag{8.72}$$

and

$$\hat{v} = \underset{v}{\operatorname{argmin}} L_c(v, u, \mu), \tag{8.73}$$

and then maximise L_c with respect to μ in an outer loop. But, since $u \in U$ and $\mu \in U$ it is possible to find μ so that

$$\hat{u} = \underset{u}{\operatorname{argmin}} L_c(v, u, \mu). \tag{8.74}$$

And it is then possible to find v and μ so that

$$\hat{u} = \underset{u}{\operatorname{argmin}} L_c(v, u, \mu), \tag{8.75}$$

and

$$\hat{v} = \underset{v}{\operatorname{argmin}} L_c(v, u, \mu), \tag{8.76}$$

and use u to maximise L_c with respect μ in an outer loop. Thus one can imagine a general algorithm for solving the problem in the following way

Algorithm 6 General KKT

1. Choose $u_0 = \{u_0^n\}_{n=1}^N$
2. For $k = 0, 1, \ldots$, do
3. Find μ_k and v_k so that

$$v_k = \underset{v}{\operatorname{argmin}} L_c(v, u_k, \mu_k) \tag{8.77}$$

   and

$$u_k = \underset{u}{\operatorname{argmin}} L_c(v_k, u, \mu_k) \tag{8.78}$$

4. Find u_{k+1} so that

$$|\nabla_\mu L_c(v_k, u_{k+1}, \mu_k)| < |\nabla_\mu L_c(v_k, u_k, \mu_k)|. \tag{8.79}$$

To perform the last step in equation (8.79), it is natural to consider Newton's method, but in principle it should be possible to use other methods. As for the augmented Lagrangian method, we will do the minimisation in equations (8.77) and (8.78) sequentially, by a Gauss-Seidel method, giving us what we have named the KKT algorithm, which we have been applying in papers A, C and D,

Algorithm 7 KKT Algorithm

1. Choose $v_0 = \{v_0^n\}_{n=1}^N$, $u_0 = \{u_0^n\}_{n=1}^N$ and $c > 0$
2. For $k = 0, 1, \ldots$, do
3. Find μ_k so that
$$u_k = \underset{u}{\operatorname{argmin}} L_c(v_k, u, \mu_k) \tag{8.80}$$
4. Find v_{k+1} so that
$$v_{k+1} = \underset{v}{\operatorname{argmin}} L_c(v, u_k, \mu_k) \tag{8.81}$$
5. For $n = 1, \ldots, N$, find u_{k+1}^n so that
$$u_{k+1}^n = u_k^n - \left(\frac{\partial e^n}{\partial u^n}\right)^{-1} e^n(v_{k+1}^n, u_k^n, u_{k+1}^{n-1}). \tag{8.82}$$

In equation (8.80), the Lagrangian multipliers are found so that $\frac{\partial L_c}{\partial u^n} = 0$ for all n. Written out in detail, equation (8.80) are the system of linear equations,

$$\frac{\partial \boldsymbol{J}^N}{\partial u^N} + (\mu^N + c \cdot e^N)^T \frac{\partial e^N}{\partial u^N} = 0, \tag{8.83}$$

and for $n = 1, \ldots, N-1$

$$\frac{\partial \boldsymbol{J}^n}{\partial u^n} + (\mu^n + c \cdot e^n)^T \frac{\partial e^n}{\partial u^n} + (\mu^{n+1} + c \cdot e^{n+1})^T \frac{\partial e^{n+1}}{\partial u^n} = 0. \tag{8.84}$$

We see that that we can easily solve this, in a similar manner as for the adjoint method, where we first solve equation (8.83) for μ^N and then for $n = N-1, \ldots, 1$ solve equation (8.84) for μ^n.

With regards to equation (8.81), we see that we can solve it recursively, time step by time step, since $\frac{\partial J}{\partial v^n} = \frac{\partial J^n}{\partial v^n}$. And the KKT method for our problems, becomes

Algorithm 8 KKT Algorithm, for our problems

1. Choose $v_0 = \{v_0^n\}_{n=1}^N$, $u_0 = \{u_0^n\}_{n=1}^N$ and $c > 0$
2. For $k = 0, 1, \ldots$, do
3. Find μ_k^N so that

$$\frac{\partial \boldsymbol{J}^N}{\partial u^N} + (\mu_k^N + c \cdot e^N)^T \frac{\partial e^N}{\partial u^N} = 0, \tag{8.85}$$

4. For $n = N-1, \ldots, 1$ find μ_k^n so that

$$\frac{\partial \boldsymbol{J}^n}{\partial u^n} + (\mu_k^n + c \cdot e^n)^T \frac{\partial e^n}{\partial u^n} + (\mu_k^{n+1} + c \cdot e^{n+1})^T \frac{\partial e^{n+1}}{\partial u^n} = 0. \tag{8.86}$$

5. For $n = 1, \ldots, N$, find v_{k+1}^n so that

$$v_{k+1}^n = \underset{v^n}{\operatorname{argmin}}\, L_c^n(v^n, u_k, \mu_k) \tag{8.87}$$

6. For $n = 1, \ldots, N$, find u_{k+1}^n so that

$$u_{k+1}^n = u_k^n - \left(\frac{\partial e^n}{\partial u^n}\right)^{-1} e^n(v_{k+1}^n, u_k^n, u_{k+1}^{n-1}). \tag{8.88}$$

8.8.1 The role of the penalty constant in the KKT algorithm

As for the augmented Lagrangian method, we make use of a penalty constant, $c > 0$, and we know from numerical experience that the method does not converge if c is too small. Furthermore, when c is greater than some value, the algorithm converges for our examples. After this critical value, where the KKT algorithm converges, the convergence rate tends to get slower, the more we increase c. It is not known how to find an optimal c, and what its role is in the optimisation. However, it may be connected to that one needs that the second order conditions for optimality to be fulfilled, as for the augmented Lagrangian method.

8.9 Other Possibilities

We have investigated the adjoint method, the augmented Lagrangian method and the KKT algorithm, and all of these methods correspond to either the way of

looking at the problem as (8.28) or as (8.30), but we have not investigated the possibility (8.29). If we consider the saddle point problem as

$$\min_u L_c(\hat{v}, u, \hat{\mu}) \quad \text{subject to} \quad \begin{cases} \hat{v} = \text{argmin}_v \, L_c(v, u, \mu) \\ \hat{\mu} = \text{argmax}_\mu \, L_c(v, u, \mu) \end{cases} \tag{8.89}$$

we may develop a method based on this. Since the derivative of L_c with respect to μ is the equation residual, and the state variables are of the same dimension as the Lagrangian multipliers, we get the algorithm

Algorithm 9 Other possibility

1. Choose $u_0 = \{u_0^n\}_{n=1}^N$
2. For $k = 0, 1, \ldots$, do
3. Find u_k and v_k so that

$$v_k = \underset{v}{\text{argmin}} \, L_c(v, u_k, \mu_k) \tag{8.90}$$

and

$$e^n(v_k, u_k) = 0, \text{ for } n = 1, \ldots N \tag{8.91}$$

4. Find μ_{k+1} so that

$$|\nabla_u L_c(v_k, u_k, \mu_{k+1})| < |\nabla_u L_c(v_k, u_k, \mu_k)|. \tag{8.92}$$

Even though it may be possible to solve the saddle point problem with this method, we are required to solve the forward problem at each iteration. This is computationally expensive, and when comparing to the KKT method we see that one iteration of the KKT method is cheaper and is the preferred one. Therefore, we have not considered this method in our work.

Chapter 9

Future Work

In this thesis we have investigated the possibility of using a saddle point formulation of the augmented Lagrangian to develop algorithms, in order to solve optimal control problems. Although the emphasis has been on solving optimal control problems which occur in reservoir engineering, we have tried to formulate the algorithms as generally as possible. We have investigated the augmented Lagrangian method introduced by Hestenes [24] and Powell [42] for this purpose. Since it in it's original form may be quite time consuming, we have looked at several possible different marching schemes, that simplifies the calculations in hope to find more efficient ways of solving the problem. The old marching scheme which has been used by Nilssen et al. in [36] and [37] has been analysed, and we have found that it cannot work for problems where the optimal value of the objective function is far from zero. To remedy this we have proposed a new marching scheme, that attempts to circumvent these shortcomings in paper D. The new marching scheme has shown to be convergent for our problems, but the convergence rate is unfortunately slow. It is possible that one could speed up the convergence of this scheme, by applying second order methods for the outer maximisation problem instead of a steepest descent method, and this could be an interesting way of further research.

Furthermore, we have proposed a new method of solving the optimal control problem, by what we have named the KKT method, where the we start with an augmented Lagrangian saddle point formulation, and try to solve the outer maximisation problem with Newton's method by changing the state variables. The strength of the method is that we do not need to solve the forward problem at each iteration. Since solving the forward problem is a very expensive in terms of computational effort, it makes one iteration of the KKT method much cheaper than for instance the adjoint method. This method has shown to be very efficient, and it looks very promising as it compares well in computational effort used, when compared to the adjoint method. We have proved this method to be convergent, for a

simplified optimal control problem, but still we have no proof of convergence for the general optimal control problem. In the future we hope to complete a proof of convergence for the general case. The KKT method uses Newton's method in the outer loop, and thus it could be interesting to look at techniques for globalising the method.

In the marching schemes, derived from the augmented Lagrangian method and in the KKT method, the penalty parameter plays a significant role, where we know that it has to be greater than some value in order to have convergence, and for the KKT method, we have found that a very large value of the parameter slows down the convergence. There seems to be a problem-dependent optimal penalty parameter, and it is of great interest to construct a method for finding this optimal value, or at least a criterion for it's optimal value.

Chapter 10

Summary of Papers

10.1 Summary of Paper A and D

Paper A: An Efficient Method for Smart Well Production Optimisation.
Authors: Daniel Chr. Doublet, Sigurd I. Aanonsen and Xue-Cheng Tai.

Paper D:Efficient Optimisation of Production from Smart Wells Based on the Augmented Lagrangian Method
Authors: Daniel Chr. Doublet, Raymond Martinsen, Sigurd I. Aanonsen and Xue-Cheng Tai.

In paper D, we consider water flood optimisation with smart well management for a two-dimensional oil-water reservoir. We enforce valve control by allocation of total injection/production amongst the different valve openings, so that our control is the percentage of total rate at each valve opening.

The problem is defined as the as it is in Brouwer and Jansen [7]. The problem is stated as an augmented Lagrangian saddle point problem, where we solve the KKT conditions for the augmented Lagrangian with the KKT method. The unconstrained optimisation problems are solved with the LBFGS method.

We provide numerical examples, where we consider three different horizontal synthetic reservoirs. For comparison, we also solve these problems with the adjoint method. We plot the objective function against the number of iterations of the KKT method and compare it to adjoint method, where the objective function is plotted against the number of times the forward problem is solved. The KKT method shows rapid convergence, and the optimal NPV found, is approximately the same as the one found with the adjoint method in all the cases we test.

Paper A is an extension of paper D. We state the production optimisation problem is stated as an optimal control problem, and we apply the KKT method to

solve it. To do the sub-minimisations of that are required in the KKT method, we use the LBFGS method as in paper D, with a line-search that ensures that the Wolfe conditions are fulfilled.

Furthermore we provide a proof of convergence for the KKT method for a simplified problem.

To deal with the additional constraints on the controls, we use projected gradients to deal with the bounds, and a technique for transforming the controls in order to eliminate the equality constraints. This technique is also described in chapter 7.

Finally we present 6 numerical examples, where we first consider a coarse discretisation with one channel. Then we refine our discretisation and increase the number of controls. We look at permeability distributions with one high permeable channel and with two high permeable channels. Our examples are inspired by those in Brouwer and Jansen [7].

As a basis of comparison we use the adjoint method, and in our examples we see that the KKT method finds the same or higher NPV than the adjoint method. Since one iteration of the KKT method is computationally cheaper than one iteration of the adjoint method, we count the number of times that linear systems are solved for both methods, in order to compare the convergence rate of the KKT method against that of the adjoint method.

Our results indicate that the KKT method converges faster than the adjoint method, when using computational effort as a measure.

10.2 Summary of Paper B

Paper B: Marching schemes for the Augmented Lagrangian Method, applied to Optimal Control Problems Occurring in Reservoir Engineering.
Authors: Daniel Chr. Doublet, Xue-Cheng Tai and Sigurd I. Aanonsen.

Paper B considers optimal control problems that occur in connection with reservoir engineering. Specifically optimisation problems, under the constraints that the reservoir flow equations are fulfilled. Examples of such problems include the production optimisation problem and the history matching problem as defined in this work. We investigate the use of the augmented Lagrangian method to these type of problems, and we analyse the marching schemes of the augmented Lagrangian method used by Nilssen et al. in [37, 36]. They are termed the Gauss-Seidel scheme and the old marching scheme. In the papers by Nilssen et al. [37, 36], the old marching scheme seems to be the better of the two.

It is shown that the old marching scheme can only be utilised for problems where the objective function is close to zero at optimum. For the problems con-

sidered by Nilssen et al. [37, 36], this was always the case which explains why it works for their problems.

For the production optimisation problem, the optimum value of the objective function cannot be zero, since we are maximising the NPV, thus the old marching scheme will not converge to the maximum NPV.

To remedy this, we propose an improvement the old marching scheme, which we call the new marching scheme, where we circumvent this problem by adding an additional term to the augmented Lagrangian functional.

Numerical examples are presented, where we consider the production optimisation problem. In the first example we consider a problem with two controls. Since we have only two controls, it enables us to plot the NPV as a function of the controls. This is valuable, as we can easily test the quality of the solution found. In the second example, the number of controls is increased. In both examples we consider a permeability field with a high permeable channel connecting the injector with the producer. For both examples we solve the problem with the adjoint method, so that we can compare the results.

For both examples we apply the Gauss-Seidel scheme, the old marching scheme and the new marching scheme. We see that the new marching scheme finds the highest value of NPV, comparable to the NPV found with the adjoint method. The Gauss-Seidel scheme, also finds an NPV close to that found with the adjoint method. However, we observe that the Gauss-Seidel method seems to be more unstable, showing a oscillatory behaviour. As our analysis indicated, the old marching scheme does not seems to be able to find as high profit as the adjoint method.

10.3 Summary of Paper C

Paper C: Efficient History Matching and Production Optimisation with the augmented Lagrangian Method.
Authors: Daniel Chr.Doublet, Sigurd I. Aanonsen and Xue-Cheng Tai

This paper considers the history matching problem and the production optimisation problem. The two problems are stated as optimal control problems and viewed as an augmented Lagrangian saddle point problem. The KKT method is used to solve these two problems.

For the history matching problem, we attempt to determine absolute permeability one the basis of pressure and saturation measurements from the wells at every time step, and seismic information of pressures and saturations at some distinct time steps.

We provide three numerical examples, from a synthetic field with one injec-

tion well and one production well, where the the difference between the three experiments is the frequency of the seismic measurements.

First the measurements are produced by running the forward model, and than adding random Gaussian noise. After generating synthetic measurements, we solve the problem with the KKT method and to compare the results, we have also solved the same problems with the adjoint method.

For all the experiments, we see that the KKT method converges, and see that the results are comparable to the results obtained with the adjoint method.

For the production optimisation problem, we consider a reservoir with two smart wells, and try to optimise the net present value, by adjusting the percentage of the total injection/production at each valve opening.

We solve the problem with the KKT method, for three different values of the penalty parameter, and we provide comparisons with the adjoint method. Again, we see that the KKT method is convergent, but the convergence rate depends on the value of the penalty parameter c.

Bibliography

[1] Shaikhan Mohammed Al-Khodhori. Smart well technologies implementation in pdo for production & reservoir management & control. *Middle East Oil Show, 9-12 June 2003, Bahrain*, 2003. SPE 81486.

[2] Eliana Arenas and Norbert Dolle. Smart waterflooding tight fractured reservoirs using inflow control valves. *SPE Annual Technical Conference and Exhibition, 5-8 October 2003, Denver, Colorado*, 2003. SPE 84193.

[3] K. Aziz and A. Settari. *Petroleum Reservoir Simulation*. Kluwer Academic Publishers, 1979.

[4] Dimitri P. Bertsekas. Combined primal-dual and penalty methods for constrained optimization. *S.J.C.*, 13:521–544, 1975.

[5] Dimitri P. Bertsekas. *Constrained Optimisation and Lagrange Multiplier Methods*. Athena Scientific, Belmont, Massachusetts, 1996.

[6] J. Frédéric Bonnans, J. Charles Gilbert, Claude Lemaréchal, and Claudia A. Sagastizábal. *Numerical Optimization, Theoretical and Practical Aspects*. Springer Verlag, 2006.

[7] D. R. Brouwer and J.-D. Jansen. Dynamic optimization of waterflooding with smart wells using optimal control theory. *SPE Journal*, 9(4):391–402, 2004.

[8] D.R. Brouwer, J.D. Jansen, and S. van der Starre. Recovery increase through water flooding with smart well technology. *SPE European Formation Damage Conference , 21-22 May, The Hague, Netherlands*, 2001. SPE 68979.

[9] Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific and Statistical Computing*, 16(5):1190–1208, 1995.

[10] Richard H. Byrd, Jorge Nocedal, and Robert B. Schnabel. Representations of quasi-newton matrices and their use in limited memory methods. *Mathematical Programming 63*, 4:129–156, 1994.

[11] T. F. Chan and X.-C. Tai. Identification of discontinuous coefficients from elliptic problems using total variation regularization. Technical Report CAM 97-35, Dept. of Math., Univeristy of California at Los Angeles, 1997.

[12] G. Chavent and J Jaffré. *Mathematical models and finite elements for reservoir simulation: single phase, multiphase and multicomponent flows through porous media*, volume 17 of *Studies in mathematics and its applications*. Amsterdam:North-Holland, 1986.

[13] G.M. Chavent, M. Dupuy, and P. Lemonnier. History mathching by the use of optimal control theory. *SPE J.*, 15, 1975.

[14] W.H. Chen, G.R. Gavalas, J.H. Seinfeld, and M.L. Wasserman. A new algorithm for automatic history matching. *SPE J.*, 1974.

[15] Zhiming Chen and Jun Zou. An augmented Lagrangian method for identifying discontinuous parameters in elliptic systems. *SIAM J. Control Optim.*, 37(3):892–910 (electronic), 1999.

[16] R. Ewing, editor. *The mathematics of reservoir simulation*. Frontiers in applied mathematics. SIAM, 1983.

[17] R. Fletcher. *Practical methods of Optimisation*. John Wiley and Sons, Chichester, 2 edition, 1987.

[18] H. Gai. Downhole flow control optimization in the worlds 1st extended reach multilateral well at wytch farm. *SPE/IADC Drilling Conference, 27 February-1 March 2001, Amsterdam, Netherlands*, February 2001. SPE 67728.

[19] Roland Glowinski. *Numerical Methods for Nonlinear Variational Problems*. Springer Verlag, 1984.

[20] Roland Glowinski and Patrick Le Tallec. *Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics*. Society for Industrial and Applied Mathematics, 1989.

[21] Ben-yu Guo and Jun Zou. An augmented Lagrangian method for parameter identifications in parabolic systems. *J. Math. Anal. Appl.*, 263(1):49–68, 2001.

[22] J. Hademard. Sur les problems aux derivés partielles et leaur signification physique. *Princeton University Bulletin*, pages 49–52, 1902.

[23] B.-O. Heimsund. *Mathematical and Numerical Methods for Reservoir Fluid Flow Simulation*. Department of Mathematics, University of Bergen, 2005.

[24] M. R. Hestenes. Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 1969.

[25] J. Holland. Genetic algorithms. *Scientific American*, 267:66–72, 1992.

[26] K. Ito, M. Kroller, and K. Kunisch. A numerical study of an augmented Lagrangian method for the estimation of parameters in elliptic systems. *SIAM J. Sci. Statist. Comput.*, 12(4):884–910, 1991.

[27] Kazufumi Ito and Karl Kunisch. The augmented Lagrangian method for parameter estimation in elliptic systems. *SIAM J. Control Optim.*, 28(1):113–136, 1990.

[28] Kazufumi Ito and Karl Kunisch. Augmented Lagrangian-SQP methods for nonlinear optimal control problems of tracking type. *SIAM J. Control Optim.*, 34(3):874–891, 1996.

[29] Yee Lo Keung and Jun Zou. Numerical identifications of parameters in parabolic systems. *Inverse Problems*, 14(1):83–100, 1998.

[30] Yee Lo Keung and Jun Zou. An efficient linear solver for nonlinear parameter identification problems. *SIAM J. Sci. Comput.*, 22(5):1511–1526 (electronic), 2000.

[31] A. Kharghoria, F. Zhang, R. Li, and Y. Jalali. Application of distributed electrical measurements and inflow control in horizontal wells under bottom-water drive. *European Petroleum Conference, 29-31 October 2002, Aberdeen, United Kingdom*, 2002. SPE 78275.

[32] K. Kunisch and X.-C. Tai. Sequential and parallel spitting methods for bilinear control problems in hilbert spaces. *SIAM Journal of Numerical Analysis*, 34(1):91–118, 1997.

[33] Martha Lien, Jan Dirk Jansen, Trond Mannseth, and Roald Brouwer. Multiscale regularization of dynamic water flood optimization. *Computational Geosciences*, 2005.

[34] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and Teller E. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[35] T. K. Nilssen, K. H. Karlsen, T. Mannseth, and X.-C. Tai. Identification of diffusion parameters in a nonlinear convection-diffusion equation using the augmented lagrangian method. *J*, 2003.

[36] T. K. Nilssen and X. C. Tai. Parameter estimation with the augmented Lagrangian method for a parabolic equation. *J. Optim. Theory Appl.*, 124(2):435–453, 2005.

[37] Trygve K. Nilssen, Trond Mannseth, and Xue-Cheng Tai. Permeability estimation with the augmented Lagrangian method for a nonlinear diffusion equation. *Comput. Geosci.*, 7(1):27–47, 2003.

[38] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Verlag, 1999.

[39] A. Ouenes, S. Bhagavan, P. Bunge, and B. Travis. Application of simulated annealing and other global optimization methods to reservoir description:myths and realities. *Proceedings of the SPE 69th Annual Technical Conference and Exhibition(New Orelans)*, March 1994. SPE28415.

[40] D. Peaceman. *Fundamentals of numerical reservoir simulation*. Elsevier scientific publishing company, 1977.

[41] Ø. Pettersen. *Grunnkurs i reservoarmekanikk*. Department of Mathematics, University of Bergen, 1990.

[42] M. J. D. Powell. A method for nonlinear constrains in minimization problems. *R. Fletcher, Editor, Academic Press New York*, 1969.

[43] W.F. Ramirez. *Application of Optimal Control Theory to Enhanced Oil Recovery*. Elsevier, 1987.

[44] Pallav Sarma, Khalid Aziz, and Louis J. Durlofsky. Implementation of adjoint solution for optimal control of smart wells. *Paper 92864-MS, in proceedings of the SPE Reservoir Simulation Symposium, 31 January-2 Feburary, The Woodlands, Texas*, 2005.

[45] M. Sen, A. Datta-Guapta, P. Stoffa, and G. Pope. Stochastic reservoir modeling using simulated annealing and gentic algorithms. *SPE Formation Evaluation*, 10:49–56, March 1995.

[46] Nigel Snaith, Richard Chia, Devarajan Narayasamy, and Kirby Schrader. Experience with operation of smart wells to maximise oil recovery from complex reservoirs. *SPE International Improved Oil Recovery Conference in Asia Pacific, 20-21 October 2003, Kuala Lumpur, Malaysia*, 2003. SPE 84855.

[47] B. Yeten, L. J. Durlofsky, and K Aziz. Optimization of nonconventional well type, location and trajectory. *SPE journal*, pages 200–210, 2003. SPE 86880.

[48] Iskander S. Zakirov, Sigurd Ivar Aanonsen, Ernest S. Zakirov, and Boris M Palatnik. Optimizing reservoir performance by allocation of well rates. *5th European conference on the Mathematics of Oil Recovery, Leoben, Austrian*, 1996.

[49] Ciyou Zhu, Richard H. Byrd, Peihuang Lu, and Jorge Nocedal. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Trans. Math. Software*, 23(4):550–560, 1997.

Part II

Included Papers

Paper A

An Efficient Method for Smart Well Production Optimisation.

AN EFFICIENT METHOD FOR SMART WELL PRODUCTION OPTIMISATION

Daniel Chr. Doublet, Sigurd I. Aanonsen and Xue-Cheng Tai

November 30, 2007

Abstract

A method for dynamic optimisation of water flooding with smart wells is presented, which finds the optimal injection and production rates for every well segment of the smart wells. We formulate the problem as a constrained optimisation problem and state this problem as an augmented Lagrangian saddle point problem, which we solve efficiently with the method. Comparisons are made with a more traditional optimal control method, based on solving the adjoint systems of equations. In the examples tested the method obtains same maximum profit, using less computational effort.

1 Introduction

As the oil resources of the world are becoming increasingly difficult to recover, it has become more important to produce existing fields as efficiently as possible and to decrease the development and operating costs. This problem is an optimisation problem where we want to maximise some profit function. In this paper we propose a method for maximising the net present value (NPV) of an oil reservoir, by reducing water production and increasing oil recovery at the same time as we are delaying water breakthrough. Optimal control theory methods has earlier been used to solve this problem in [3], [12], [15], [16], using the adjoint method. However, when solving the problem by this method, it is required to solve the state equations exactly for each new estimate of the control variables, which is computationally expensive.

In this work, we formulate the optimisation problem as an augmented Lagrangian saddle point problem, and present a method for solving it, by solving the Karush-Kuhn-Tucker (KKT) conditions for the augmented Lagrangian functional. The KKT conditions are solved sequentially, avoiding to solve the state equations exactly for each new estimate of the control variables, and thereby reducing the computational cost for finding a new

estimate of the controls. Although the state equations are not fulfilled during the optimisation procedure, they will be so at convergence. Preliminary results has been presented earlier in [7] and [6].

The use of the augmented Lagrangian functional for constrained optimisation, with, was introduced by Hestenes in [9] and has been applied to optimal control problems later in [10], [11]. In this paper we make use of the augmented Lagrangian functional, but the solution approach is somewhat different than the one used in [10] and [11].

A proof of convergence is provided for a quadratic optimisation problem with linear constraints, showing that it is possible to find a penalisation parameter that guaranties convergence in the quadratic case. Several numerical examples are presented, showing the effectiveness of the method and we also present comparisons with the more traditional adjoint method.

2 Problem Formulation

We consider a rectangular, heterogeneous, two-dimensional, two-phase (oil and water) reservoir with no-flow boundaries inspired by the model of Brouwer and Jansen [3]. The model is horizontal such that gravitational effects can be ignored. There is a smart well injector along the left edge of the reservoir, and a smart well producer along the right edge of the reservoir, as shown in figure 1. The two smart wells have several valve openings, indicated by black dots, such that the injection and the production can be controlled individually at each of the valve openings. Initially the reservoir is completely oil saturated, and at the start of operation water is injected in the valve openings at the left hand side of the reservoir, whilst we are producing from the valve openings at the right hand side of the reservoir. In the beginning only oil is produced, but after a certain time we will start to produce both oil and water, and finally only water is produced. There is a profit associated with production of oil. However when producing both oil and water simultaneously we must remove the water from the oil, resulting in a cost associated with water production. The total production rate is fixed, and we assume that we inject at the same rate as we produce. We have the possibility to control the profit by adjusting the percentage of total injection/production in each of the valve openings. Consequentially our objective is to maximise the NPV from an oil reservoir by adjusting the individual injection and production rates, in the valve openings, at distinct times. The reservoir flow equations for two phase flow without gravity are,

$$-\phi\frac{\partial S_o}{\partial t} - \nabla\cdot\left(\kappa(x)\lambda_o(S_o)\nabla p_o\right) = q_o(x), \tag{1}$$

and

$$-\phi\frac{\partial S_w}{\partial t} - \nabla\cdot\left(\kappa(x)\lambda_w(S_w)\nabla p_w\right) = q_w(x), \tag{2}$$

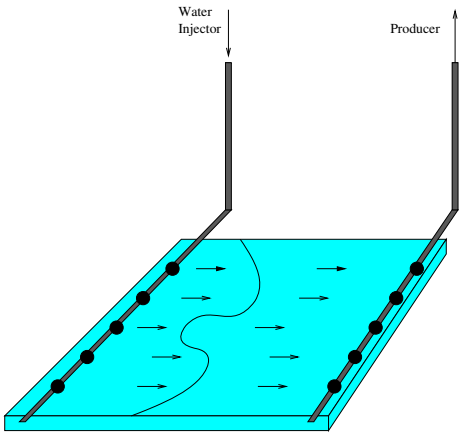

Figure 1: Reservoir.

where x is position, t is time, ϕ is porosity, κ is absolute permeability, $\lambda_\alpha = \kappa_{r\alpha}/\mu_\alpha$ is the phase mobility, where $\kappa_{r\alpha}$ is relative permeability and μ_α is viscosity, p_α is pressure, S_α is saturation and the well term, q_α, is flow rate per unit volume, where the subscripts α denotes the fluid phase, o or w. The reservoir is assumed to be fully saturated so that

$$S_w + S_o = 1. \tag{3}$$

From now on, and in the rest of this paper, we will denote the water saturation as S, and the oil saturation as $1 - S$. Furthermore we assume that the capillary pressure, $p_c = p_o - p_w$, is zero so that

$$p_w = p_o = p. \tag{4}$$

In the remaining part of this paper, the pressure is denoted by p. In addition we have no-flow boundaries so that,

$$\frac{\partial p}{\partial \vec{n}} = 0, \tag{5}$$

where $\vec{n}$ is the unit normal vector to the boundary. The relative permeabilities are defined by the Corey models

$$\kappa_{ro} = \kappa^\star_{ro}\left(\frac{1 - S - S_{or}}{1 - S_{or} - S_{wr}}\right)^{e_o} \tag{6}$$

and

$$\kappa_{rw} = \kappa^\star_{rw}\left(\frac{S - S_{wr}}{1 - S_{wr} - S_{or}}\right)^{e_w}, \tag{7}$$

where e_o and e_w are the Corey exponents, $\kappa^\star_{ro}$ and $\kappa^\star_{rw}$ are the endpoint permeabilities, and S_{or} and S_{wr} are the residual saturations, for oil and water respectively.

The equations (1) and (2) are discretised using a standard cell centred grid, with the scheme given by Aziz and Sattari in [1], with upstream weighting and using backward Euler to approximate the time derivative. From the discretisation we get a discrete time model of the two-phase conservation equations

$$e^n(\hat{v}^n, \hat{u}^n, \hat{u}^{n-1}) = 0, \quad \text{for} \quad \text{n} = 1, \dots \text{N}, \tag{8}$$

where $e^n = \{e_o^n, e_w^n\}^T$ is the residual column vector corresponding to the discretised equations of (1) and (2) , the superscript n denotes the discrete time step, N is the total number of time steps, $\hat{u}^n = \{p^n, S^n\}$ is a column vector consisting of the pressures and the water saturations in all the grid cells at time step n, and $\hat{v}^n$ is the column vector consisting of the control variables, at time step n, which elements are related to the water injection and liquid production rates in the different valve openings of the wells.

Instead of using a well model, we will directly control water injection and liquid production at each valve opening, such that the valve openings of the wells are treated as source terms. We assume that the total water injection rate equals the total liquid production rate, and we denote the total injection/production rate by V.

Letting v_i^n denote the percent of total injection or production in valve opening i at time step n, we have the relations

$$\sum_{i=1}^{N_{inj}} v_i^n = 1, \quad \text{for} \quad n = 1, \dots, N, \tag{9}$$

and

$$\sum_{i=1+N_{inj}}^{N_{prod}+N_{inj}} v_i^n = 1, \quad \text{for} \quad n = 1, \dots, N, \tag{10}$$

where N_{inj} is the number of valve openings in the injector and N_{prod} is the number of valve openings in the producer. All the valve openings from number 1 to number N_{inj} are assumed to belong to the injection well, while the valve openings from number $1+N_{inj}$ to number $N_{prod}+N_{inj}$ are assumed to belong to the production well. Moreover, the control variables must also satisfy the following inequality constraints, for $n = 1, \dots, N$,

$$0 \leq v_i^n \leq 1, \quad \text{for} \quad i = 1, \dots, N_{inj} + N_{prod}. \tag{11}$$

Since only water is injected, the liquid rate equals the water rate for the valve openings in the injector. Thus we have that, for $n = 1, \dots, N$,

$$q_{wi}^n = v_i^n V, \quad \text{for} \quad i = 1, \dots, N_{inj}, \tag{12}$$

where q_{wi}^n is the water rate at injection valve opening i at time step n. In the valve openings of the producer, however, the liquid rate equals the sum of the water and oil rates so that, for $n = 1, \dots, N$,

$$q_{wi}^n + q_{oi}^n = -v_i^n V, \quad \text{for} \quad i = 1 + N_{inj}, \dots, N_{prod} + N_{inj}, \tag{13}$$

where q_{wi}^n and q_{oi}^n are the water and oil rates, respectively, at production valve opening i at time step n. The different phase production rates can be expressed as functions of the liquid rate and the fractional flow at the well segment so that

$$q_{wi}^n = -\frac{\lambda_{wi}^n}{\lambda_{wi}^n + \lambda_{oi}^n} \, v_i^n V \quad \text{for} \quad i = 1 + N_{inj}, \ldots, N_{prod} + N_{inj}, \tag{14}$$

and

$$q_{oi}^n = -\frac{\lambda_{oi}^n}{\lambda_{oi}^n + \lambda_{wi}^n} \, v_i^n V, \quad \text{for} \quad i = 1 + N_{inj}, \ldots, N_{prod} + N_{inj}, \tag{15}$$

where λ_{wi}^n and λ_{oi}^n are the water and oil mobilities, respectively, in the grid cell containing well segment i and at time step n.

2.1 Profit Function

Our aim is to maximise net present value, by controlling the individual valve opening rates during the entire production period. The net present value (NPV), J, is given as

$$J(\hat{v}, \hat{u}) = \sum_{n=1}^{N} J^n(\hat{v}^n, \hat{u}^n), \tag{16}$$

with

$$J^n(\hat{v}^n, \hat{u}^n) = \Delta x \Delta y h \left[\sum_{i=1+N_{inj}}^{N_{prod}+N_{inj}} \frac{-I_w \cdot q_{wi}^n - I_o \cdot q_{oi}^n}{(1 + b/100)^{t^n}} \right] \Delta t^n, \tag{17}$$

where the constants I_o and I_w are, respectively, the revenue of oil produced and the cost of water produced per volume expressed in $\$/m^3$, b is the annual interest rate expressed in %, Δx and Δy are the dimensions of the grid cells in, respectively, horizontal and vertical direction, Δt^n is the size of the n'th time step, and $t^n = \sum_{i=1}^{n} \Delta t^i$ is the time expressed in years at time step n. Since the production rates of water and oil, q_{wi}^n and q_{oi}^n, are less than or equal to zero, I_o is a positive constant and I_w is a negative constant. The objective is to maximise the function (16) by adjusting the percentage of total injection and production in each of the individual valve openings of, respectively, the injection well and the production well. It must however be noted that if we at some point produce so much water that J^n in equation (17) is negative, then we set J^n to zero, since we in practice will stop the production if this happens. This is done so that we maximise the profit over the period of time when we produce, while still letting the profitable production period be unknown.

2.2 A Saddle Point Formulation

Let us define the control variables at time step n as $\hat{v}^n = \{v_i^n\}_{i=1}^{N_{inj}+N_{prod}}$, and let $\hat{v} = \{\hat{v}^n\}_{n=1}^N$ and $\hat{u} = \{\hat{u}^n\}_{n=1}^N$. We can now formulate our problem as a constrained minimisation problem where we want to solve

$$\min_{\hat{v},\hat{u}} -J(\hat{v},\hat{u}) \tag{18}$$

subject to

$$e^n(\hat{v}^n,\hat{u}^n,\hat{u}^{n-1}) = 0, \quad \text{for } n = 1,\ldots,N. \tag{19}$$

The corresponding Lagrangian is

$$L(\hat{v},\hat{u},\eta) = \sum_{n=1}^{N} L^n(\hat{v}^n,\hat{u}^n,\hat{u}^{n-1},\eta^n), \tag{20}$$

where

$$L^n(\hat{v}^n,\hat{u}^n,\hat{u}^{n-1},\eta^n) = -J^n(\hat{v}^n,\hat{u}^n) + \eta^{nT} e^n(\hat{v}^n,\hat{u}^n,\hat{u}^{n-1}), \tag{21}$$

and $\eta = \{\eta^n\}_{n=1}^N$ are the Lagrangian multipliers. Furthermore, we define the augmented Lagrangian by

$$L_c(\hat{v},\hat{u},\eta) = \sum_{n=1}^{N} L_c^n(\hat{v}^n,\hat{u}^n,\hat{u}^{n-1},\eta^n), \tag{22}$$

where

$$L_c^n(\hat{v}^n,\hat{u}^n,\hat{u}^{n-1},\eta^n) = L^n(\hat{v}^n,\hat{u}^n,\hat{u}^{n-1},\eta^n) + \frac{c}{2} e^n(\hat{v}^n,\hat{u}^n,\hat{u}^{n-1})^T e^n(\hat{v}^n,\hat{u}^n,\hat{u}^{n-1}), \tag{23}$$

and $c > 0$ is a penalisation constant. It is known that the solution of (18), (19) is a saddle point of (22) for a sufficiently large penalisation parameter c, see for example Bonnans et al. [2] pp. $275 - 279$. for proof. Given that the set of constraint gradients is linearly independent at the solution of (18), (19), the following conditions hold at the solution of (18), (19), see e.g. Nocedal and Wright [13].

2.2.1 Karush Kuhn Tucker (KKT) conditions

Suppose that $\{\hat{v}^\star,\hat{u}^\star\}$ is a solution of (18). Then there exists a set of vectors $\eta^\star$ such that the following conditions hold at the point $\{\hat{v}^\star,\hat{u}^\star\}$,

$$\begin{aligned} \nabla_{\hat{u}^n} L(\hat{v}^\star,\hat{u}^\star,\eta^\star) &= 0, \\ \nabla_{\hat{v}^n} L(\hat{v}^\star,\hat{u}^\star,\eta^\star) &= 0, \\ \nabla_{\eta^n} L(\hat{v}^\star,\hat{u}^\star,\eta^\star) &= 0, \end{aligned} \tag{24}$$

for $n = 1, \ldots, N$. Since the last of these conditions yields $e^n = 0$, for $n = 1, \ldots, N$, it is easy to see that the KKT conditions also hold for the augmented Lagrangian functional

$$\begin{aligned} \nabla_{\hat{u}^n} L_c(\hat{v}^\star, \hat{u}^\star, \eta^\star) &= 0, \\ \nabla_{\hat{v}^n} L_c(\hat{v}^\star, \hat{u}^\star, \eta^\star) &= 0, \\ \nabla_{\eta^n} L_c(\hat{v}^\star, \hat{u}^\star, \eta^\star) &= 0, \end{aligned} \tag{25}$$

for $n = 1, \ldots, N$. In order to find the solution of (18), we can thus solve the KKT system (24) or equivalently (25). In the recent years this problem has been solved using the adjoint method, see e.g. Brouwer and Jansen [3], which has given satisfactory results. However, when solving this problem by the adjoint method, we are required to solve the forward problem, eq.(8), exactly each time we want to find a new gradient. Solving the forward problem is time consuming, and we present here a method for solving this problem, without the need to solve the forward problem exactly, at each iteration. Thus we propose here a algorithm to solve (18), (19) by solving the system of equations (25).

3 Optimisation Method

Letting the subscript k be the outer iteration counter, we propose the following algorithm to solve the KKT conditions (25).

Algorithm 1 KKT Optimisation Algorithm

1. Choose $\hat{v}_0 = \{\hat{v}_0^n\}_{n=1}^N$, $\hat{u}_0 = \{\hat{u}_0^n\}_{n=1}^N$, $c > 0$. For $k = 1, 2, \ldots$ do:
2. Find η_k^N such that

$$\nabla_{\hat{u}^N} L_c^N(\hat{v}_{k-1}^N, \hat{u}_{k-1}^N, \hat{u}_{k-1}^{N-1}, \eta_k^N) = 0. \tag{26}$$

3. For $n = N-1, N-2, \ldots, 1$, find η_k^n, such that

$$\nabla_{\hat{u}^n} L_c^{n+1}(\hat{v}_{k-1}^{n+1}, \hat{u}_{k-1}^{n+1}, \hat{u}_{k-1}^n, \eta_k^{n+1}) + \nabla_{\hat{u}^n} L_c^n(\hat{v}_{k-1}^n, \hat{u}_{k-1}^n, \hat{u}_{k-1}^{n-1}, \eta_k^n) = 0. \tag{27}$$

4. Then, for $n = 1, 2, \ldots, N$, find $\hat{v}_k^n$ such that

$$\hat{v}_k^n = arg\min_{\hat{v}^n} L_c^n(\hat{v}^n, \hat{u}_{k-1}^n, \hat{u}_{k-1}^{n-1}, \eta_k^n). \tag{28}$$

5. Then update $\hat{u}_{k-1} = \{\hat{u}_{k-1}^n\}_{n=1}^N$ by

$$\hat{u}_k^n = \hat{u}_{k-1}^n - \left(\nabla_{\eta^n \hat{u}^n} L_c(\hat{v}_k, \hat{u}_{k-1}, \eta_k)\right)^{-1} \nabla_{\eta^n} L_c(\hat{v}_k, \hat{u}_{k-1}, \eta_k). \tag{29}$$

After finding η_k from equations (26) and (27) in the algorithm, the first of the KKT conditions are fulfilled. By studying the augmented Lagrangian given by equation (22) and (23) we observe that $\nabla_{\hat{u}^N} L_c = \nabla_{\hat{u}^N} L_c^N$ and that $\nabla_{\hat{u}^n} L_c = \nabla_{\hat{u}^n} L_c^n + \nabla_{\hat{u}^n} L_c^{n+1}$, so that $\nabla_{\hat{u}^N} L_c$ is dependent on $\hat{v}^N, \hat{u}^N, \hat{u}^{N-1}$ and η^N and $\nabla_{\hat{u}^n} L_c$, for $n = 1, 2, \ldots, N-1$, is dependent on $\hat{v}^{n+1}$, $\hat{v}^n$, $\hat{u}^{n+1}$, $\hat{u}^n$, $\hat{u}^{n-1}$, η^{n+1} and η^n. Thus the first of the KKT conditions can be solved exactly for η if we start by finding η^N such that $\nabla_{\hat{u}^N} L_c = 0$ and then, for $n = N-1, N-2, \ldots, 1$, we find η^n such that $\nabla_{\hat{u}^n} L_c = 0$. Furthermore we observe that, when calculating $\hat{v}_k$ from equation (28) in the algorithm above, we find $\hat{v}_k$ such that

$$\nabla_{\hat{v}^n} L_c(\hat{v}_k, \hat{u}_{k-1}, \eta_k) = 0,$$

for $n = 1, \ldots, N$. Thus the second of the KKT conditions are fulfilled after performing the calculations in equation (28). Finally in equation (29) of the algorithm, we are solving the third of the KKT conditions by a Newton-iteration. In this way, the third of the KKT conditions is fulfilled at convergence, and since we are enforcing the first and the second of the KKT conditions in equation (26), (27) and (28) the solution is found at convergence. Using this algorithm on our particular problem, we get, omitting here the arguments

Algorithm 2 KKT Optimisation Algorithm, in Detail

1. Choose $\hat{v}_0 = \{\hat{v}_0^n\}_{n=1}^N$, $\hat{u}_0 = \{\hat{u}_0^n\}_{n=1}^N$, $c > 0$. For $k = 1, 2, \ldots$ do:
2. Find η_k^N such that

$$-\frac{\partial J^N}{\partial \hat{u}^N} + \left(\eta_k^N + c \cdot e^N\right)^T \frac{\partial e^N}{\partial \hat{u}^N} = 0, \tag{30}$$

3. For $n = N-1, N-2, \ldots, 1$, find η_k^n, such that

$$-\frac{\partial J^n}{\partial \hat{u}^n} + \left(\eta_k^n + c \cdot e^n\right)^T \frac{\partial e^n}{\partial \hat{u}^n} + \left(\eta_k^{n+1} + c \cdot e^{n+1}\right)^T \frac{\partial e^{n+1}}{\partial \hat{u}^n} = 0, \tag{31}$$

4. Then, for $n = 1, 2, \ldots, N$,we will find $\hat{v}_k^n$ such that

$$\hat{v}_k^n = arg \min_{\hat{v}^n} \left[-J^n + \left(\eta^n + \frac{c}{2} e^n\right)^T e^n \right] \tag{32}$$

5. And finally update the state variables $\hat{u}^n$ by

$$\hat{u}_k^n = \hat{u}_{k-1}^n - \left(\frac{\partial e^n}{\partial \hat{u}^n}\right)^{-1} e^n. \tag{33}$$

When finding η^n, for $n = N, \ldots, 1$ in equations (30) and (31) of the algorithm we need to solve a linear system of equations, and we do this with the GMRES method, as described in for example Golub and van Loan [8] or Saad [14]. For the minimisation in equation (32), we use the LBFGS method which is a quasi-Newton method. For more information on this method see Byrd et al. [5], [4]. Finally, for the update in equation (33), we use as in equations (30) and (31), the GMRES method to solve the linear system of equations.

3.1 The Quadratic Problem with Linear Constraints

In order to understand the convergence properties of the optimisation method proposed, it makes sense to investigate the performance of the algorithm on a simplified problem. And we will thus consider a quadratic optimal control problem with linear constraints. Let us consider the simplified problem

$$\min_{\hat{v},\hat{u}} J(\hat{v},\hat{u}) = \min_{\hat{v},\hat{u}} \frac{1}{2}\hat{v}^T H_{\hat{v}\hat{v}}\hat{v} + \frac{1}{2}\hat{u}^T H_{\hat{u}\hat{u}}\hat{u} + \hat{v}^T H_{\hat{u}\hat{v}}\hat{u} + g_{\hat{v}}^T\hat{v} + g_{\hat{u}}^T\hat{u}, \quad (34)$$

subject to

$$K(\hat{v},\hat{u}) = N\hat{v} + B\hat{u} - b = 0, \quad (35)$$

where $H_{\hat{v}\hat{v}} \in \mathbb{R}^{n_c \times n_c}$, $H_{\hat{u}\hat{u}} \in \mathbb{R}^{n_s \times n_s}$, $B \in \mathbb{R}^{n_s \times n_s}$, $H_{\hat{u}\hat{v}} \in \mathbb{R}^{n_c \times n_s}$, $N \in \mathbb{R}^{n_c \times n_s}$, $g_{\hat{v}} \in \mathbb{R}^{n_c}$, $g_{\hat{u}} \in \mathbb{R}^{n_s}$, $b \in \mathbb{R}^{n_s}$ and $K \in \mathbb{R}^{n_s}$. Furthermore, B is assumed to be non-singular. Let us define the augmented Lagrangian as

$$L_c(\hat{v},\hat{u}) = J(\hat{v},\hat{u}) + K(\hat{v},\hat{u})^T\eta + \frac{c}{2}K(\hat{v},\hat{u})^T K(\hat{v},\hat{u}), \quad (36)$$

where $\eta \in \mathbb{R}^{n_s}$ is the Lagrangian multipiler. Now, the KKT conditions of optimality for L_c gives that

$$\begin{bmatrix} H_{\hat{v}\hat{v}} + cN^TN & H_{\hat{u}\hat{v}} + cN^TB & N^T \\ H_{\hat{u}\hat{v}}^T + cB^TN & H_{\hat{u}\hat{u}} + cB^TB & B^T \\ N & B & 0 \end{bmatrix} \begin{bmatrix} \hat{v} \\ \hat{u} \\ \eta \end{bmatrix} = - \begin{bmatrix} g_{\hat{v}}^T - cN^Tb \\ g_{\hat{u}}^T - cB^Tb \\ -b \end{bmatrix}. \quad (37)$$

The second order conditions of optimality gives that

$$z^T \begin{bmatrix} H_{\hat{v}\hat{v}} & H_{\hat{u}\hat{v}} \\ H_{\hat{u}\hat{v}}^T & H_{\hat{u}\hat{u}} \end{bmatrix} z > 0, \quad (38)$$

for all $z \in N(A)$, where $A = \begin{pmatrix} N & B \end{pmatrix}$. Solving the problem (34)-(35) with our algorithm gives the following iterations

$$\begin{aligned} \hat{u}_{k+1} &= -B^{-1}(-b + N\hat{v}_k) \\ \eta_{k+1} &= -B^{-T}(g_{\hat{u}}^T - cB^Tb + (H_{\hat{u}\hat{v}}^T + cB^TN)\hat{v}_k + (H_{\hat{u}\hat{u}} + cB^TB)\hat{u}_{k+1}) \\ \hat{v}_{k+1} &= -(H_{\hat{v}\hat{v}} + cN^TN)^{-1}[g_{\hat{v}}^T - cN^Tb + (H_{\hat{u}\hat{v}} + cN^TB)\hat{u}_{k+1} + N^T\eta_{k+1}]. \end{aligned} \quad (39)$$

We may simplify this iteration by substituting $\hat{u}_{k+1}$ in the second equation by the first, giving

$$\begin{aligned}\eta_{k+1} = -B^{-T}(g_{\hat{u}}^T - cB^T b + (H_{\hat{u}\hat{v}}^T + cB^T N)\hat{v}_k \\ - (H_{\hat{u}\hat{u}} + cB^T B)B^{-1}(-b + N\hat{v}_k)).\end{aligned} \tag{40}$$

Furthermore we substitute $\hat{u}_{k+1}$ and η_{k+1} in the third equation by the first and the second equation, giving

$$\begin{aligned}\hat{v}_{k+1} = -(H_{\hat{v}\hat{v}} + cN^T N)^{-1}[g_{\hat{v}}^T - cN^T b - (H_{\hat{u}\hat{v}} + cN^T B)B^{-1}(-b + N\hat{v}_k) \\ - N^T B^{-T}(g_{\hat{u}}^T - cB^T b + (H_{\hat{u}\hat{v}}^T + cB^T N)\hat{v}_k \\ - (H_{\hat{u}\hat{u}} + cB^T B)B^{-1}(-b + N\hat{v}_k))].\end{aligned} \tag{41}$$

By simple algebraic manipulation this algorithm becomes

$$\begin{aligned}\hat{u}_{k+1} &= -B^{-1}(-b + N\hat{v}_k) \\ \eta_{k+1} &= -B^{-T}(g_{\hat{u}}^T + H_{\hat{u}\hat{u}}B^{-1}b + (H_{\hat{u}\hat{v}}^T - (H_{\hat{u}\hat{u}})B^{-1}N)\hat{v}_k) \\ \hat{v}_{k+1} &= (H_{\hat{v}\hat{v}} + cN^T N)^{-1}[-g_{\hat{v}}^T - H_{\hat{u}\hat{v}}B^{-1}b + N^T B^{-T} g_{\hat{u}}^T \\ &\quad + N^T B^{-T} H_{\hat{u}\hat{u}} B^{-1} b + (H_{\hat{u}\hat{v}} B^{-1} N + N^T B^{-T} H_{\hat{u}\hat{v}}^T \\ &\quad + cN^T N - N^T B^{-T} H_{\hat{u}\hat{u}} B^{-1} N)\hat{v}_k].\end{aligned} \tag{42}$$

From (42), we observe that $\hat{u}_{k+1}$ and η_{k+1} depend on $\hat{v}_k$ and are independent of $\hat{u}_k$ and η_k, so that the spectrum of the algorithm equals the spectrum of $\hat{v}_{k+1}$. Thus, when defining M by

$$\begin{aligned}M = (H_{\hat{v}\hat{v}} + cN^T N)^{-1}(H_{\hat{u}\hat{v}} B^{-1} N + N^T B^{-T} H_{\hat{u}\hat{v}}^T \\ + cN^T N - N^T B^{-T} H_{\hat{u}\hat{u}} B^{-1} N),\end{aligned} \tag{43}$$

the algorithm is convergent if the spectral radius of M

$$\rho(M) = |\max(eig(M)| < 1. \tag{44}$$

Lemma 1 The spectral radius of M has the following properties

1. $\rho(M) < 1$
2. For $\forall \epsilon > 0, \quad \rho(M) \in (-\epsilon, 1)$ for c large enough.

Proof:

Denoting a eigenvalue of M by α and the corresponding eigenvector by w, we have that

$$(H_{\hat{u}\hat{v}}B^{-1}N + N^TB^{-T}H_{\hat{u}\hat{v}}^T + cN^TN - N^TB^{-T}H_{\hat{u}\hat{u}}B^{-1}N)w = \alpha(H_{\hat{v}\hat{v}} + cN^TN)w. \quad (45)$$

Multiplying (45) on the right by w^T, and can reformulate as

$$w^TH_{\hat{v}\hat{v}}w - 2w^TH_{\hat{u}\hat{v}}B^{-1}Nw + w^TN^TB^{-T}H_{\hat{u}\hat{u}}B^{-1}Nw = (1-\alpha)(w^TH_{\hat{v}\hat{v}}w^T + c||Nw||^2). \quad (46)$$

It follows from the second order optimality conditions (38) that

$$w^TH_{\hat{v}\hat{v}}w - 2w^TH_{\hat{u}\hat{v}}B^{-1}Nw + w^TN^TB^{-T}H_{\hat{u}\hat{u}}B^{-1}Nw > 0, \quad (47)$$

so that (46) gives that

$$(1-\alpha)(w^TH_{\hat{v}\hat{v}}w^T + c||Nw||^2) > 0. \quad (48)$$

According to Finsler's lemma, see for example Bonnans et al. [2], p. 285 , the second order optimality conditions (38) gives that

$$w^TH_{\hat{v}\hat{v}}w^T + c||Nw||^2 > 0, \quad (49)$$

and it follows that

$$\alpha < 1. \quad (50)$$

Now, we look at the second part. Let $\epsilon > 0$, and assume by contradiction that
$\alpha \leq -\epsilon$. Suppose, that for a sequence of $c \to \infty$, there exists a sequence of corresponding unit eigenvectors $w_c \to w \neq 0$ so that (46) holds

$$w_c^TH_{\hat{v}\hat{v}}w_c - 2w_c^TH_{\hat{u}\hat{v}}B^{-1}Nw_c + w_c^TN^TB^{-T}H_{\hat{u}\hat{u}}B^{-1}Nw_c = (1-\alpha)(w_c^TH_{\hat{v}\hat{v}}w_c^T + c||Nw_c||^2). \quad (51)$$

$$\geq (1+\epsilon)(w_c^TH_{\hat{v}\hat{v}}w_c^T + c||Nw_c||^2) \quad (52)$$

If we divide equation (51) by c, and let c tend to its limit, we get that

$$(1+\epsilon)||Nw||^2 \leq 0, \quad (53)$$

Hence it follows that $Nw = 0$, and equation (51) gives when c tends to its limit

$$w^TH_{\hat{v}\hat{v}}w \geq (1+\epsilon)(w^TH_{\hat{v}\hat{v}}w^T), \quad (54)$$

which yields the desired contradiction.

By Lemma 1, we can always choose a sufficiently large c such that $\rho(M) \in (-1, 1)$ and thereby verifying condition (44) such we are ensured convergence.

3.2 Comparison With The Adjoint Method

To compare the results of the KKT algorithm, we use an optimal control theory method, the adjoint method, used previously in [3], [12], [15] and [16], to solve the production optimisation problem. Letting, as for the KKT algorithm, the subscript k be the outer iteration counter, the adjoint algorithm is as follows.

Algorithm 3 The Adjoint Algorithm

1. Choose $\hat{v}_0 = \{\hat{v}_0^n\}_{n=1}^N$, $\hat{u}_0^0$ and $c > 0$. For $k = 1, 2, \ldots$ do:
2. For $n = 1, 2, \ldots, N$, find $\hat{u}_k^n$ such that
$$e^n(\hat{v}_{k-1}^n, \hat{u}_k^n, \hat{u}_k^{n-1}) = 0. \tag{55}$$
3. Find η_k^N such that
$$-\frac{\partial J^N}{\partial \hat{u}^N}(\hat{v}_{k-1}^N, \hat{u}_k^N) + \eta_k^{N\,T}\frac{\partial e^N}{\partial \hat{u}^N}(\hat{v}_{k-1}^N, \hat{u}_k^N, \hat{u}_k^{N-1}) = 0. \tag{56}$$
4. For $n = N-1, N-2, \ldots, 1$, find η_k^n, such that
$$\begin{aligned} -\frac{\partial J^n}{\partial \hat{u}^n}(\hat{v}_{k-1}^n, \hat{u}_k^n) + \eta_k^{n\,T}\frac{\partial e^n}{\partial \hat{u}^n}(\hat{v}_{k-1}^n, \hat{u}_k^n, \hat{u}_k^{n-1}) \\ + \eta_k^{n+1\,T}\frac{\partial e^{n+1}}{\partial \hat{u}^n}(\hat{v}_{k-1}^{n+1}, \hat{u}_k^{n+1}, \hat{u}_k^n) = 0. \end{aligned} \tag{57}$$
5. Finally we find the gradient of $-J$ as
$$\begin{aligned} -\frac{dJ^n}{d\hat{v}^n}(\hat{v}_{k-1}^n) &= \frac{\partial L_c}{\partial \hat{v}^n}(\hat{v}_{k-1}^n, \hat{u}_k^n) \\ &= -\frac{\partial J^n}{\partial \hat{v}^n}(\hat{v}_{k-1}^n, \hat{u}_k^n) + \eta_k^{n\,T}\frac{\partial e^n}{\partial \hat{v}^n}(\hat{v}_{k-1}^n, \hat{u}_k^n, \hat{u}_k^{n-1}) = 0, \end{aligned} \tag{58}$$
and use this gradient in a the LBFGS algorithm to find $\hat{v}_k = \{\hat{v}_k^n\}_{n=1}^N$.

Equation (55) in the algorithm is in fact the same as solving the forward problem, and when solving the non-linear systems of equations in (55), we use Newton's method so that we find $\hat{u}_k^n$ as

- Choose $\hat{u}_{k,0}^n$. For $l = 1, 2, \ldots$ do until convergence: For $n = 1, 2, \ldots, N$ do:
$$\hat{u}_{k,l}^n = \hat{u}_{k,l-1}^n - \left(\frac{\partial e^n}{\partial \hat{u}^n}(\hat{v}_{k-1}^n, \hat{u}_{k,l-1}^n)\right)^{-1} e^n(\hat{v}_{k-1}^n, \hat{u}_{k,l-1}^n, \hat{u}_{k,l}^{n-1}), \tag{59}$$
where we solve the linear system of equations with the GMRES method.

In order to find η^n for the adjoint algorithm we need to solve a linear system of equations in (56) and in (57), for which we use the GMRES method.

3.3 Estimation of Cost per Iteration

Let us denote the number of grid cells by M_c, and let $M_w = N_{inj} + N_{prod}$ denote the total number of valve openings of injection and production type. Now we investigate the computational cost of our two algorithms.

For one iteration of the KKT algorithm:

- N linear systems of equations of size $2 \cdot M_c$ (equation (30) and (31).
- N non-linear systems of equations of size M_w(equation (32))
- N linear systems of equations of size $2 \cdot M_c$(equation (33)).

In practice, the computational time required to perform the calculations in (32) in the KKT algorithm is negligible compared to that of (30), (31) and (33), since $M_w << M_c$. Thus the dominating factor is the time required to solve $2 \cdot N$ linear systems of equations of size $2 \cdot M_c$.

For one iteration of the adjoint method:

- N linear systems of equations of size $2 \cdot M_c$(equation (56) and (57))
- N non-linear systems of equations of size $2 \cdot M_c$(equation (55))
- plus some calculations needed to find an approximation to the second derivative in the outer LBFGS loop.

For the adjoint method, the calculation of the derivative and the approximation to the Hessian in (58) is negligible in comparison with the work done in (55), (56) and (57). In (56) and (57) we are solving N linear systems of equations of size $2 \cdot M_c$, and in (55) the computational time depends upon how many iterations of Newton's method which is required in order to solve the forward problem. For each iteration of Newton's method, we must solve N linear systems of equations of size $2 \cdot M_c$. Our experience shows that one needs to do between 3 to 5 iterations of Newton's algorithm in order to reach convergence, giving a total of $4N$ to $6N$ linear systems of equations of size $2 \cdot M_c$ to solve in each outer loop of the algorithm. From this, one sees that the KKT algorithm is from two to three times faster per iteration compared to the adjoint algorithm. Furthermore it must be noted here that by one iteration of the adjoint algorithm we mean one iteration of the algorithm as it is stated here.

3.4 The additional Constraints on the Control Variables

In addition to the constraints given in (8), we have other equality constraints given in (9) and (10), restated here

$$\sum_{i=1}^{N_{inj}} v_i^n = 1, \quad \text{for} \quad n = 1, \ldots, N, \tag{60}$$

and

$$\sum_{i=1+N_{inj}}^{N_{prod}+N_{inj}} v_i^n = 1, \quad \text{for} \quad n = 1, \ldots, N. \tag{61}$$

In addition, we also have inequality constraints given in (11),

$$0 \leq v_i^n \leq 1, \quad \text{for} \quad i = 1, \ldots, N_{inj} + N_{prod}, \tag{62}$$

for $n = 1 \ldots N$.

A method of handling the linear equality constraints (60) and (61), could be to do the optimisation with respect to the variables $v_1^n, v_2^n, \ldots, v_{N_{inj}-1}^n$ and $v_{N_{inj}+1}^n, v_{N_{inj}+2}^n, \ldots, v_{N_{inj}+N_{prod}-1}^n \quad \text{for} \quad n = 1, \ldots, N$. Then we can calculate the remaining variables using the linear equality constraints (60) and (61) so that

$$v_{N_{inj}}^n = 1 - \sum_{i=1}^{N_{inj}-1} v_i^n \quad \text{for} \quad n = 1, \ldots, N \tag{63}$$

and

$$v_{N_{inj}+N_{prod}}^n = 1 - \sum_{i=1+N_{inj}}^{N_{prod}+N_{inj}-1} v_i^n \quad \text{for} \quad n = 1, \ldots, N. \tag{64}$$

One can ensure that the inequality constraints (62) are fulfilled for the variables $v_1^n, v_2^n, \ldots, v_{N_{inj}-1}^n$ and $v_{N_{inj}+1}^n, v_{N_{inj}+2}^n, \ldots, v_{N_{inj}+N_{prod}-1}^n \quad \text{for} \quad n = 1, \ldots, N$ by using a projected method in the optimisation, but for the variables $v_{N_{inj}}^n$ and $v_{N_{prod}+N_{inj}}^n$ for $n = 1, \ldots, N$, there is however no easy way of ensuring that the inequality constraints (62) are fulfilled.

To overcome this difficulty, we will instead use a different approach, where we define a set of $(N_{inj} + N_{prod} - 2) \cdot N$ functions

$$0 \leq \xi_1^n, \ldots, \xi_{N_{inj}+N_{prod}-2}^n \leq 1, \quad \text{for} \quad n = 1 \ldots N, \tag{65}$$

and denote $\xi^n = \{\xi_i^n\}_{i=1}^{N_{inj}+N_{prod}-2}$. Now the v^n are given in terms of ξ^n by the relations,

$$\begin{aligned}
v_1^n &= \xi_1^n \\
v_2^n &= (1 - \xi_1^n)\xi_2^n \\
v_3^n &= (1 - \xi_1^n)(1 - \xi_2^n)\xi_3^n \\
&\vdots \\
v_{N_{inj}-1}^n &= \xi_{N_{inj}-1}^n \prod_{i=1}^{N_{inj}-2} (1 - \xi_i^n) \\
v_{N_{inj}}^n &= \prod_{i=1}^{N_{inj}-1} (1 - \xi_i^n),
\end{aligned} \tag{66}$$

and

$$\begin{aligned} v^n_{1+N_{inj}} =& \xi^n_{N_{inj}} \\ v^n_{2+N_{inj}} =& (1-\xi^n_{N_{inj}})\xi^n_{1+N_{inj}} \\ v^n_{3+N_{inj}} =& (1-\xi^n_{N_{inj}})(1-\xi^n_{1+N_{inj}})\xi^n_{2+N_{inj}} \\ & \vdots \\ v^n_{N_{prod}+N_{inj}-1} =& \xi^n_{N_{prod}+N_{inj}-2} \prod_{i=1}^{N_{prod}+N_{inj}-3} (1-\xi^n_i) \\ v^n_{N_{prod}+N_{inj}} =& \prod_{i=1}^{N_{prod}+N_{inj}-2} (1-\xi^n_i). \end{aligned} \tag{67}$$

Now we observe that, when defining the rates as in (66) and (67) , (60) and (61) hold for all

$$0 \le \xi^n_1, \ldots, \xi^n_{N_{inj}+N_{prod}-2} \le 1. \tag{68}$$

Using ξ^n instead of v^n, we have a optimisation problem without the linear equality constraints. However the inequality constraints in (65) are transformed into the inequality constraints in (68), which we handle with a projected method.

To find the derivative, we have by the chain rule that for $n = 1, \ldots, N$,

$$\frac{\partial L^n_c}{\partial \xi^n_i} = \sum_{j=1}^{N_{inj}} \frac{\partial L^n_c}{\partial v^n_j}\frac{\partial v^n_j}{\partial \xi^n_i}, \quad \text{for} \quad 0 \le i \le N_{inj}-1, \tag{69}$$

and

$$\frac{\partial L^n_c}{\partial \xi^n_i} = \sum_{j=1+N_{inj}}^{N_{inj}+N_{prod}} \frac{\partial L^n_c}{\partial v^n_j}\frac{\partial v^n_j}{\partial \xi^n_i}, \quad \text{for} \quad N_{inj} \le i \le N_{inj}+N_{prod}-2. \tag{70}$$

The derivative $\frac{\partial L^n_c}{\partial v^n_j}$, we already know, and the other derivative is given by

$$\frac{\partial v^n_j}{\partial \xi^n_i} = \begin{cases} 0 & \text{if } i > j \\ \prod_{k=1}^{j-1}(1-\xi^n_k) & \text{if } i = j, \text{ and } j, i < N_{inj} \\ -\frac{\xi^n_j}{1-\xi^n_i}\prod_{k=1}^{j-1}(1-\xi^n_k) & \text{if } i < j, \text{ and } j, i < N_{inj} \\ -\frac{1}{1-\xi^n_i}\prod_{k=1}^{j-1}(1-\xi^n_k) & \text{if } i < j = N_{inj} \end{cases}$$

3.5 The additional Constraints on the State Variables

Our state variables consists of pressures and saturations in all the grid cells. On the saturation there are some additional inequality constraints, stating that the saturations cannot be less than the residual saturation,

$$S_{wr} \le S_i \le 1 - S_{or}. \tag{71}$$

Exp.	Type	Grid	Sim. time	Time steps	# param.	c
1	one channel	5×5	2 years	64	640	$1 \cdot 10^8$
2	one channel	15×15	2 years	64	1920	$1.25 \cdot 10^7$
3	two channels	15×15	2 years	64	1920	$1 \cdot 10^7$
4	two channels	15×15	2 years	64	1920	$1 \cdot 10^7$
5	one channel	45×45	5 years	171	15390	$4 \cdot 10^6$
6	two channels	45×45	5 years	149	13410	$5 \cdot 10^6$

Table 1: The numerical experiments

κ_{ro}	κ_{rw}	e_o	e_w	S_{or}	S_{wr}	μ_o	μ_w	ϕ
1.0	0.1	2.0	1.5	0.2	0.2	0.5	0.5	0.2

Table 2: Reservoir simulator constants–equal for all experiments.

When (29) is performed, in the KKT algorithm, the constraints (71) the constraints may be violated. If so is the case, the saturations are set to either S_{wr} or $1 - S_{or}$, depending on which constraint that is violated. That is, we do as follows

1. Find $\hat{u}_k^n = \{p_k^n, S_k^n\}$ from (29)
2. For $i = 1, 2, \ldots, M_c$ do:
3. If $S_{ki}^n < S_{wr}$
   $S_{ki}^n = S_{wr}$.
4. If $S_{ki}^n > 1 - S_{or}$
   $S_{ki}^n = S_{or}$.

4 Numerical Examples

In this section we present numerical examples of profit optimisation done with the KKT method. Herein we also present comparisons with the more traditional adjoint method.

In all of our examples we use synthetic reservoir models inspired by the models of Brouwer and Jansen [3] with dimensions $450m \times 450m \times 10m$. Along the left hand side of the reservoir there is one injection well, with one valve opening in each grid cell along the well. We have the possibility to control the injection at each of the individual valve openings at each time step. Similarly, there is one production well along the right hand side of the reservoir, with one valve opening in each grid cell along the producer. For the producer we also have the the possibility to control the individual production rates at each of the valve openings at each time step. As an initial guess for our rates, we use uniform injection and production, such

that the individual injection and production rates are equal at all the grid cells which are penetrated by a well. For the KKT method we also need an initial guess for the state variables u. We find this initial u by solving the forward problem for the initial rates, and using the resulting state variables, u, as our initial guess for u. Initially the water saturation is the residual water saturation of 0.2 everywhere in the reservoir, and the initial pressure is 190 bar, i.e. $19 \cdot 10^6$ Pa, in the entire reservoir. Other reservoir simulator constants are given in table 2.

For the revenue of oil produced, we set $I_o = 80$ \$$/m^3$ and set the cost associated with water production to $I_w = -20$ \$$/m^3$. Although the revenue constant is very low comparing with the prices of today, it is set deliberately to this value since we wish to make a comparison with the study done by [3]. The annual interest rate is set to 20% in equation (17).

Six experiments were performed as listed in Table 1. The permeability fields applied are plotted in figure 2. The first experiment is a small, simple test case with 25 grid cells. The next 3 experiments are coarse versions of example 1, 2, and 3 in [3], while the last two corresponds to the model sized used in [3]. One pore volume of water is injected in all cases, but the total rate is less in the last two experiments giving a longer total simulation time

The value of the penalty constant c in the augmented Lagrangian functional (22), for the six different experiments, is given in table 1. This constant is found experimentally for each specific experiment, as the one that seems to be best fitted. It is our experience that if one uses a too high value of the constant c, the KKT algorithm converges slower. On the other hand, if one uses a too small value for c, the constraints may not converge to zero such that the algorithm does not converge.

Since the equation constraint is only fulfilled at convergence for the KKT method, it is not straigth forward to compare the convergence behaviour of the KKT method with the adjoint method. In order to facilitate this comparison, we solve after each iteration of the KKT method the forward problem using the latest set of control variables. The state variables when solving the forward problem given the rates v_k will be denoted $\tilde{u}_k$. That is $\tilde{u}_k = \{u | e^n(v_k^n, u^n, u^{n+1}) = 0, \text{ for n} = 1, \ldots, \text{N}\}$. Correspondingly, $\tilde{J}_k = J(v_k, \tilde{u}_k)$ as opposed to $J_k = J(v_k, u_k)$, which is the objective function calculated from a current iteration of the KKT method.

For the different experiments, we plot $J_k = J(v_k, u_k)$, and $\tilde{J}_k$ versus the number of iterations of the KKT algorithm. We also provide plots of the development of $\sqrt{\sum_{n=1}^{N} ||e^n(v_k, u_k)||^2/(2 \cdot M_c N)}$ to show the convergence of the residual for the various experiments.

Furthermore, to compare the efficiency of the KKT method with the adjoint method, we also plot the objective function against the number of times that we solve N linear systems of equations of size $2 \cdot M_c$.

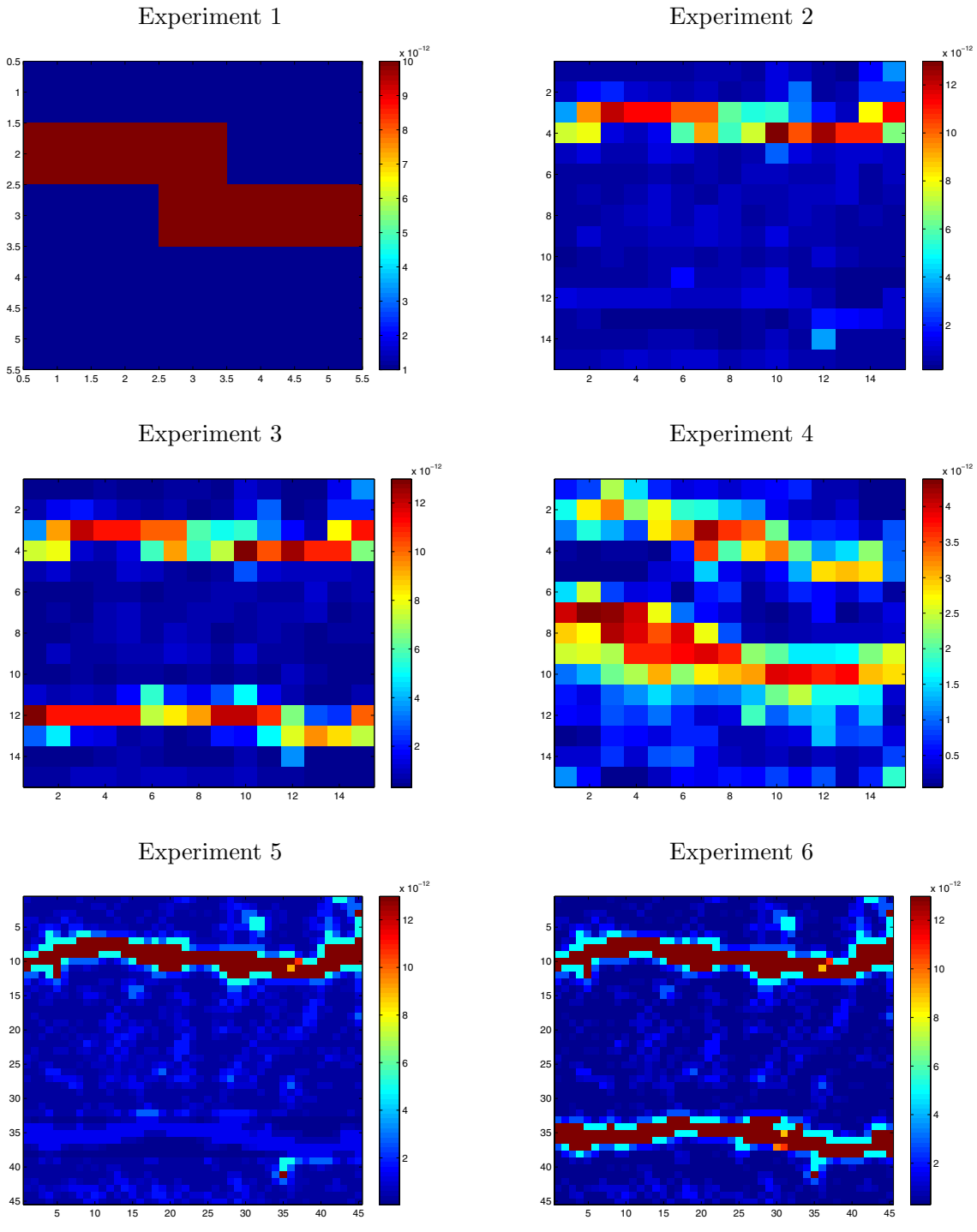

Figure 2: Permeability fields, given in m^2.

4.1 Results

The behaviour of the KKT algorithm for all the experiments are shown in Figs. 3–12. Comparison with the adjoint method is shown for the first four examples.

Notice that because the initial guess is taken from a forward model run, with an equation residual given by the convergence criteria of the forward model, the equation residual will always start at a low value. During the iterations, the equation constraint is not fulfilled, so it increases initially, before it converges to a low value as the KKT iterations proceed. Note that the equation residual converges faster than the NPV in all cases. Also, when the equation residual approaches zero, $J_k = J(v_k, u_k)$ approaches $\tilde{J}_k$ as expected, and it seems that these become approximately equal at a value of $\sqrt{\sum_{n=1}^{N} ||e^n(v_k, u_k)||^2/(2 \cdot M_c N)}$ approximately equal to 10^{-3} in all cases. For comparison, the convergence criteria used in the forward model corresponds to a value of $\sqrt{\sum_{n=1}^{N} ||e^n(v_k, u_k)||^2/(2 \cdot M_c N)}$ equal to $7 \cdot 10^{-4}$.

We also see from the figures that the optimal profit found with the KKT method is approximately the same as found with the adjoint method. However, the computational effort is less for the KKT method. Notice that the NPV does not change initially for the adjoint method. This is due to the line search done in the LBFGS optimisation routine, which is used by the adjoint method. Although not clearly seen because of the scale, there will be several linear solves without any increase in the adjoint NPV for every iteration, corresponding to the average number of iterations applied in the outer (Newton) iteration loop in each time step of the forward solver. In these examples there is a large period with little changes at the end of the simulations, where only 1 or 2 Newton iterations are required. Thus, the gain in computational speed with the KKT method may be even larger in other cases.

The final allocated relative rates found with the KKT method is shown in Figs. 13–18. Comparison with rates obtained with the adjoint method is shown for the first 4 experiments. The injection rates are shown in the top sub-figures, and the production rates are shown in bottom sub-figures. Time steps are on the horizontal axis, and well segments are on the vertical axis. Note that the plots end earlier than what is indicated in Table 1. this is because it is not profitable to continue production. From these figures we see that the optimal rates distribution for the two methods are similar, but not the same, suggesting that there are several rates distributions giving the same optimal profit, corresponding to several maxima for the objective function.

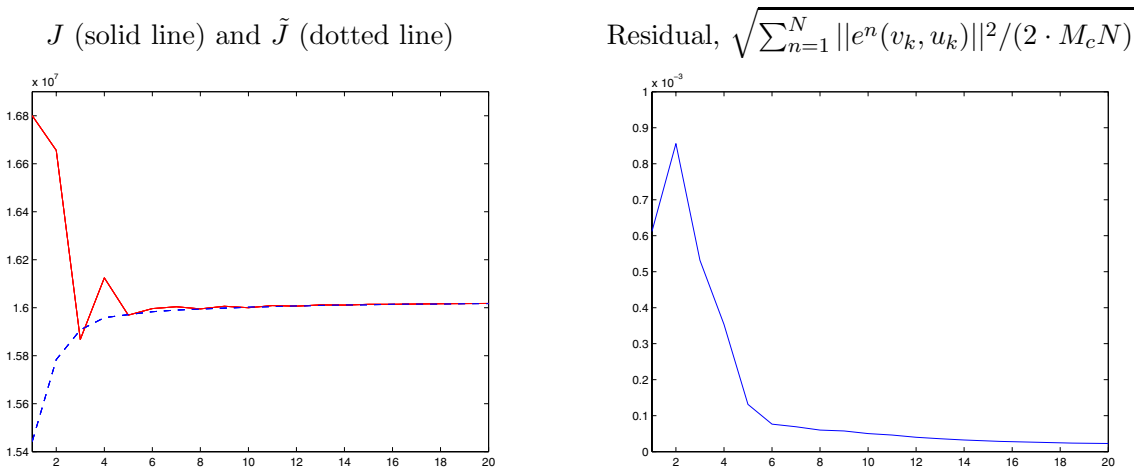

Figure 3: Objective function and equation residual for the 20 first 20 iterations of the KKT algorithm. Experiment 1.

5 Conclusions

We have presented a method for solving a NPV maximisation problem based on solving the Karush Kuhn Tucker equations for the augmented Lagrangian functional. In the examples tested, the KKT method finds approximately the same NPV value as the adjoint method with less computational effort. The performance of the KKT method is dependent on the value of the penalty parameter, c, but in the examples tested we have observed that the penalty parameter is approximately the same for reservoirs with the same number of grid cells. This suggest that, once it has been found, we do not need to change it further, when the permeability field is changed in a closed-loop profit optimisation process.

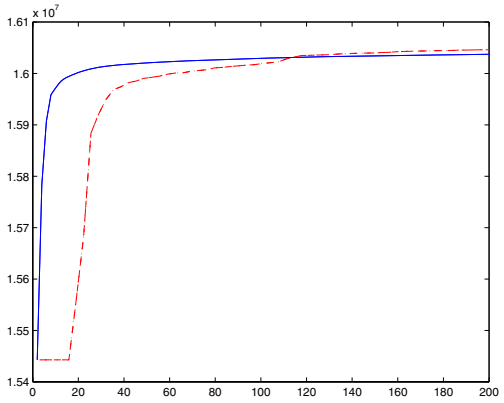

Figure 4: Development of the NPV for experiment 1. The NPV for the adjoint method is plotted as a dashed line and the true NPV, $J(v_m, \tilde{u}_m)$ for the KKT method is shown as a solid line. We have \$ on the vertical axis and the number of times N linear systems of equations of size $2 \cdot M_c$ is solved, on the horizontal axis.

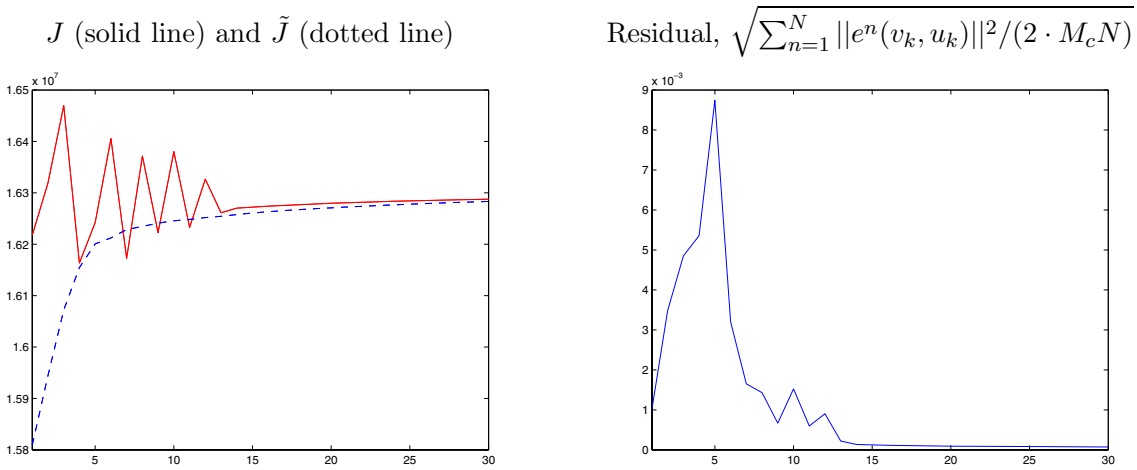

Figure 5: Objective function and equation residual for the 20 first 20 iterations of the KKT algorithm. Experiment 2.

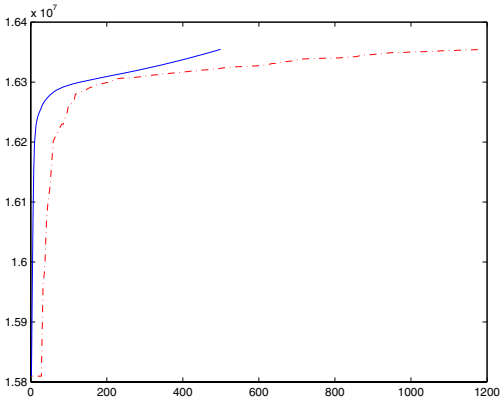

Figure 6: Development of the NPV for experiment 2. The NPV for the adjoint method is plotted as a dashed line and the true NPV, $J(v_m, \tilde{u}_m)$ for the KKT method is shown as a solid line. We have \$ on the vertical axis and the number of times N linear systems of equations of size $2 \cdot M_c$ is solved, on the horizontal axis.

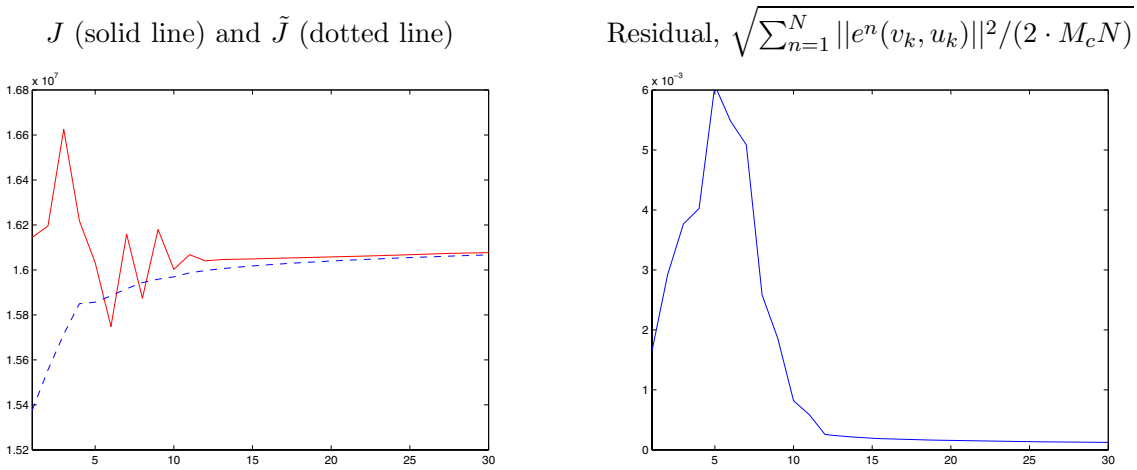

Figure 7: Objective function and equation residual for the 20 first 20 iterations of the KKT algorithm. Experiment 3.

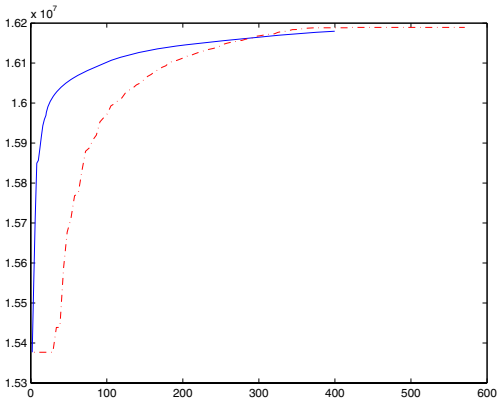

Figure 8: Development of the NPV for experiment 3. The NPV for the adjoint method is plotted as a dashed line and the true NPV, $J(v_m, \tilde{u}_m)$ for the KKT method is shown as a solid line. We have \$ on the vertical axis and the number of times N linear systems of equations of size $2 \cdot M_c$ is solved, on the horizontal axis.

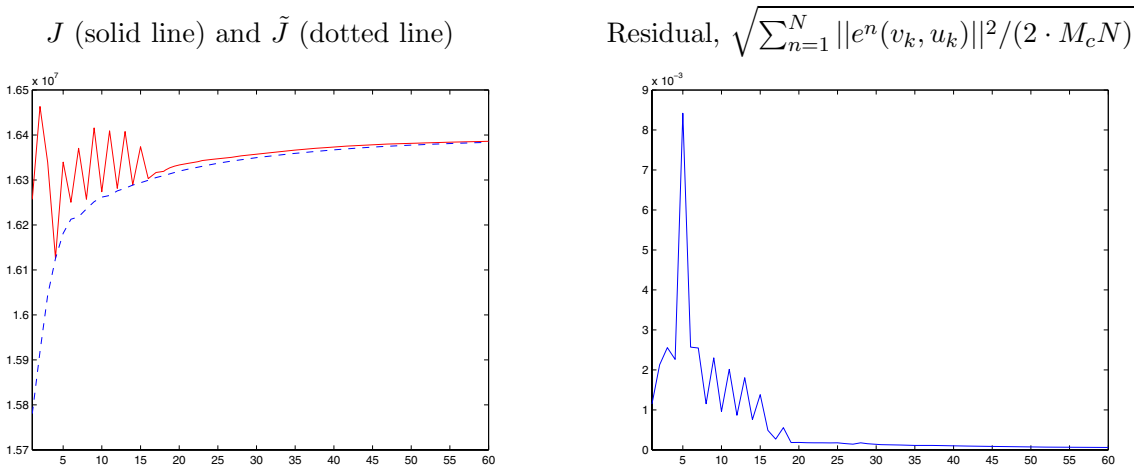

Figure 9: Objective function and equation residual for the 20 first 20 iterations of the KKT algorithm. Experiment 4.

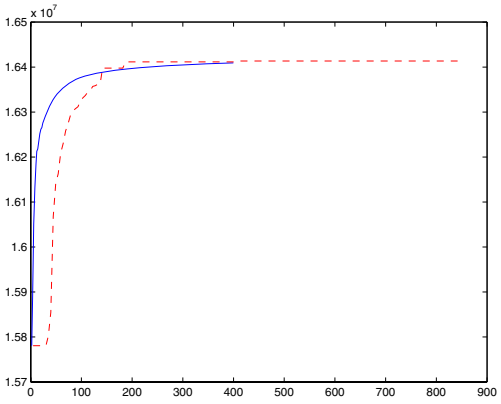

Figure 10: Development of the NPV for experiment 4. The NPV for the adjoint method is plotted as a dashed line and the true NPV, $J(v_m, \tilde{u}_m)$ for the KKT method is shown as a solid line. We have \$ on the vertical axis and the number of times N linear systems of equations of size $2 \cdot M_c$ is solved, on the horizontal axis.

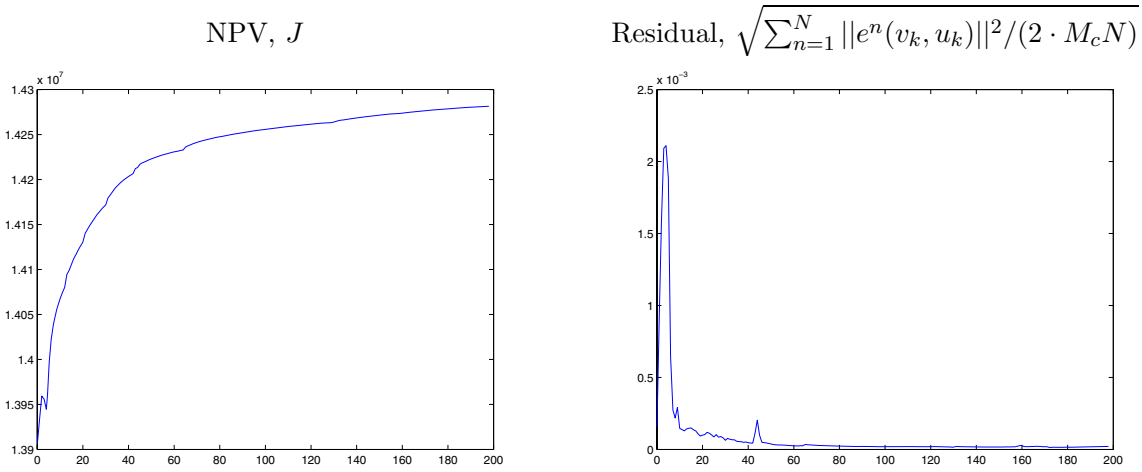

Figure 11: Objective function and equation residual vs. KKT iterations. Experiment 5.

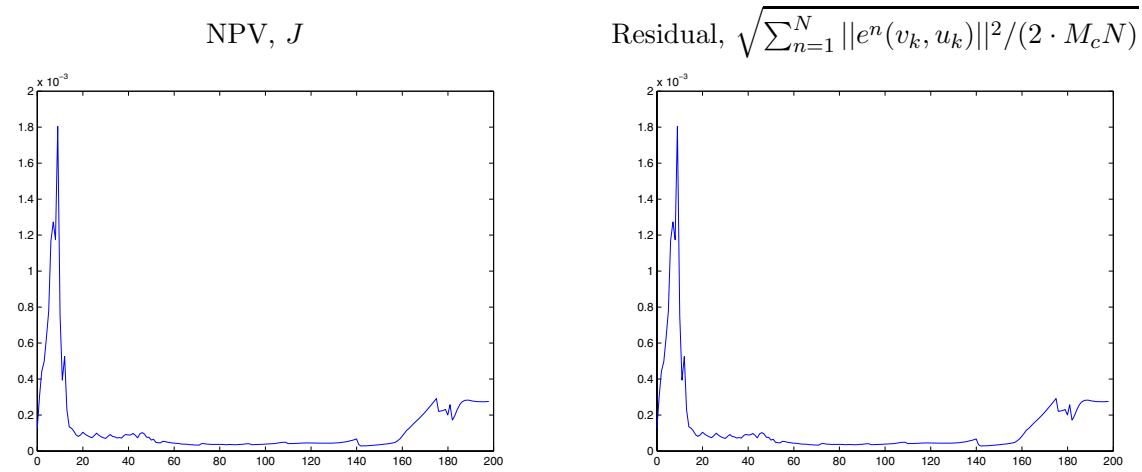

Figure 12: Objective function and equation residual vs. KKT iterations. Experiment 6.

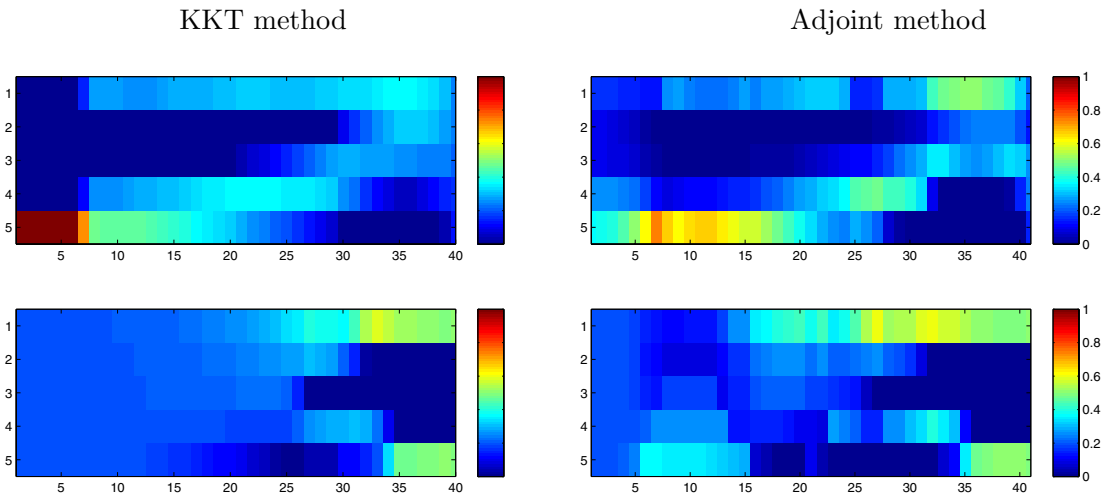

Figure 13: Final estimated relative rate allocation for experiment 1. Injection well (top) and production well (bottom). Horizontal axis: discrete time steps.Vertical axis: Well segments.

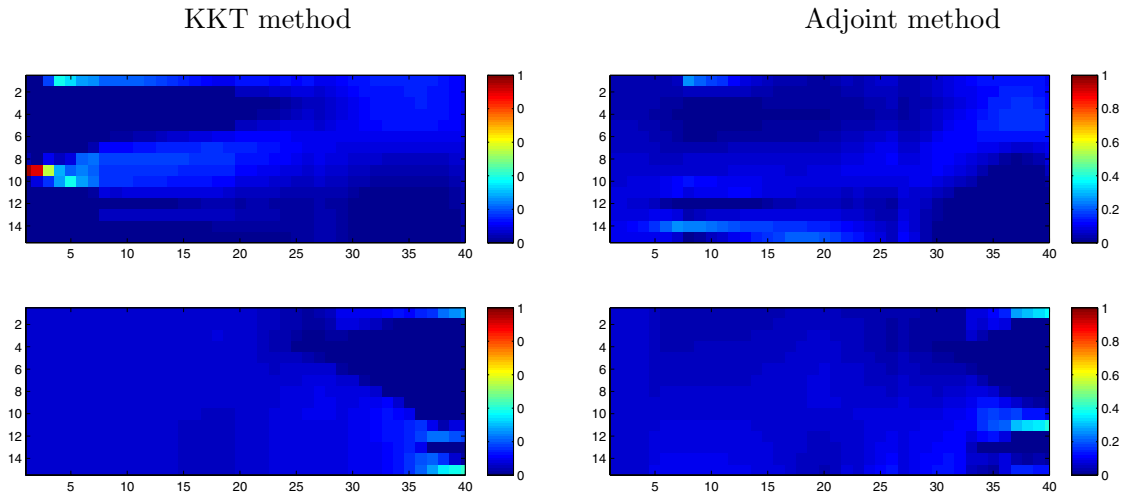

Figure 14: Final estimated relative rate allocation for experiment 2. Injection well (top) and production well (bottom). Horizontal axis: discrete time steps.Vertical axis: Well segments.

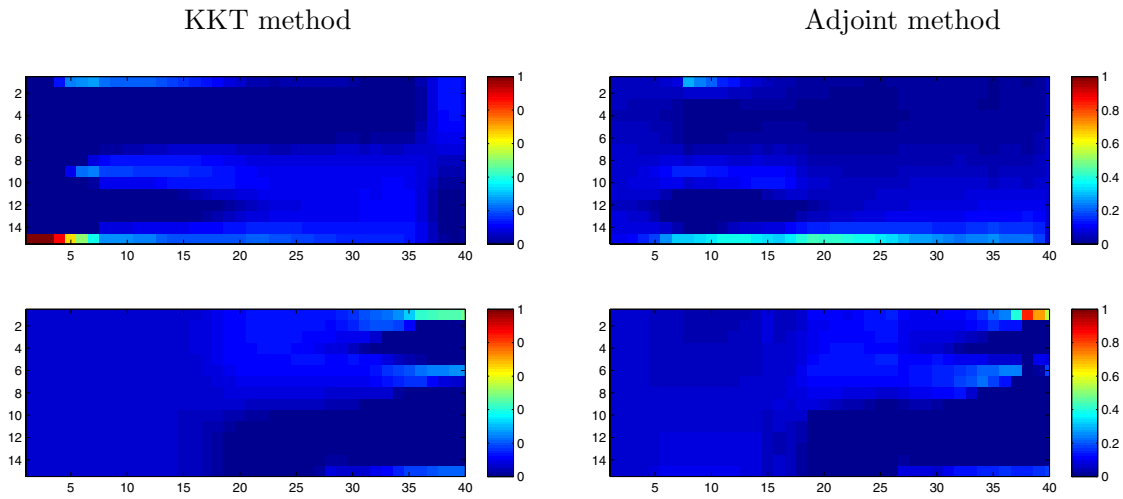

Figure 15: Final estimated relative rate allocation for experiment 3. Injection well (top) and production well (bottom). Horizontal axis: discrete time steps.Vertical axis: Well segments.

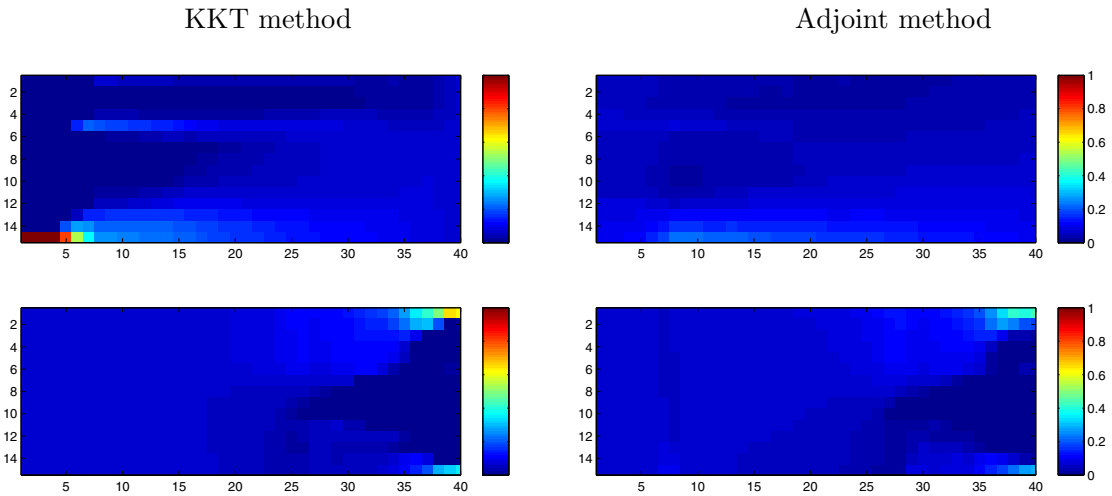

Figure 16: Final estimated relative rate allocation for experiment 4. Injection well (top) and production well (bottom). Horizontal axis: discrete time steps.Vertical axis: Well segments.

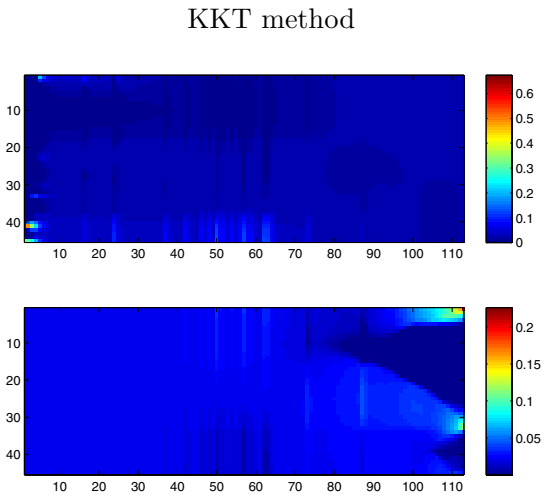

Figure 17: Final estimated relative rate allocation for experiment 5. Injection well (top) and production well (bottom). Horizontal axis: discrete time steps.Vertical axis: Well segments.

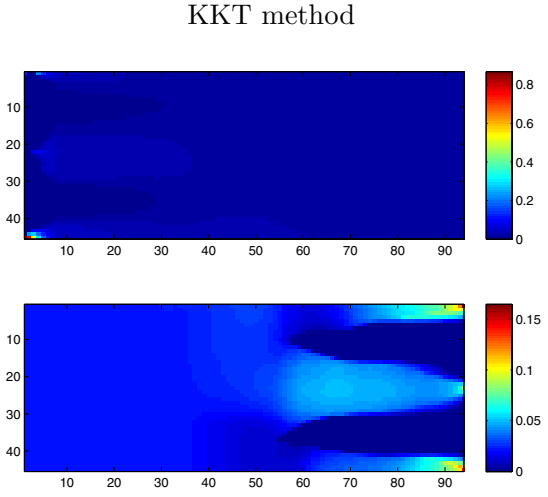

Figure 18: Final estimated relative rate allocation for experiment 6. Injection well (top) and production well (bottom). Horizontal axis: discrete time steps.Vertical axis: Well segments.

6 Acknowledgements

This work is financed by the Norwegian Research Council, Petromaks programme, project No. 163383/S30, and we are grateful for the support. Furthermore, we want to thank Raymond Martinsen for providing the code of the forward simulator.

References

[1] K. Aziz and A. Settari. *Petroleum Reservoir Simulation*. Kluwer Academic Publishers, 1979.

[2] J. Frédéric Bonnans, J. Charles Gilbert, Claude Lemaréchal, and Claudia A. Sagastizábal. *Numerical Optimization, Theoretical and Practical Aspects*. Springer Verlag, 2006.

[3] D. R. Brouwer and J.-D. Jansen. Dynamic optimization of waterflooding with smart wells using optimal control theory. *SPE Journal*, 9(4):391–402, 2004.

[4] Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific and Statistical Computing*, 16(5):1190–1208, 1995.

[5] Richard H. Byrd, Jorge Nocedal, and Robert B. Schnabel. Representations of quasi-newton matrices and their use in limited memory methods. *Mathematical Programming 63*, 4:129–156, 1994.

[6] D. C. Doublet, S. I. Aanonsen, and X.-C. Tai. Efficient history matching and production optimisation with the augmented lagrangian method. *Presented at SPE Reservoir Simulation Symposium, 26-28 February 2007, Houston, Texas, U.S.A.*, 2007.

[7] D. C. Doublet, R. Martinsen, S. I. Aanonsen, and X.-C. Tai. Efficient optimisation of production from smart wells based on the augmented lagrangian method. *Presented at European Conferance on the Mathematics of Oil Recovery X, Amsterdam, The Netherlands, september*, 2006.

[8] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 1996.

[9] M. R. Hestenes. Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 1969.

[10] Kazufumi Ito and Karl Kunisch. The augmented lagrangian method for parameter estimation in elliptic systems. *SIAM Journal of Control and Optimization*, 28(1):113–136, 1990.

[11] K. Kunisch and X.-C. Tai. Sequential and parallel spitting methods for bilinear control problems in hilbert spaces. *SIAM Journal of Numerical Analysis*, 34(1):91–118, 1997.

[12] M. Lien, R. Brouwer, J. D. Jansen, and T. Mannseth. Multiscale regularization of flooding optimization for smart field management. *Paper 99728-MS, in proceedings of the Intelligent Energy Conference and Exhibition, 11-13 April, Amsterdam, The Netherlands*, 2006.

[13] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Verlag, 1999.

[14] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Yousef Saad, 2000.

[15] Pallav Sarma, Khalid Aziz, and Louis J. Durlofsky. Implementation of adjoint solution for optimal control of smart wells. *Paper 92864-MS, in proceedings of the SPE Reservoir Simulation Symposium, 31 January-2 Feburary, The Woodlands, Texas*, 2005.

[16] Iskander S. Zakirov, Sigurd Ivar Aanonsen, Ernest S. Zakirov, and Boris M Palatnik. Optimizing reservoir performance by allocation of well rates. *5th European conference on the Mathematics of Oil Recovery, Leoben, Austrian*, 1996.

Paper B

Marching Schemes for the Augmented Lagrangian Method.

A NEW MARCHING SCHEME OF THE AUGMENTED LAGRANGIAN METHOD

Daniel C. Doublet, Xue-C. Tai and Sigurd I. Aanonsen

November 30, 2007

Abstract

We consider optimal control problems which occur in connection with reservoir engineering. The problem is formulated a constrained optimisation problem, and investigate the use of marching schemes of the augmented Lagrangian method to solve the problem. Furthermore we propose a new marching scheme, which is an improvement of the existing ones. Examples are provided, where the marching schemes are used for dynamic optimisation of water flooding with smart wells. In the examples tested, the new marching scheme performs better than the other schemes.

1 Introduction

In this paper we investigate the use of marching schemes of the augmented Lagrangian to solve optimal control problems. Firstly we define the type of optimal control problem, which we will consider here. Then we will state the augmented Lagrangian method for this problem. Further on we will discuss the use of a marching scheme used in [20] and [19] for solving our type of optimal control problem. Then we state two new alternative marching schemes, and finally we apply these marching schemes to a specific problem where we present numerical results.

1.1 An Optimal Control Problem

We consider a rectangular, heterogeneous, two-dimensional, two-phase (oil and water) reservoir, with no-flow boundaries which is horizontal such that gravitational effects can be ignored. The reservoir flow equations are,

$$-\phi\frac{\partial S_o}{\partial t} - \nabla\cdot\Big(\kappa(x)\lambda_o(S_o)\nabla p_o\Big) = q_o(x), \tag{1}$$

and

$$-\phi\frac{\partial S_w}{\partial t} - \nabla\cdot\Big(\kappa(x)\lambda_w(S_w)\nabla p_w\Big) = q_w(x), \tag{2}$$

where x is position, t is time, ϕ is porosity, κ is absolute permeability, $\lambda_\alpha = \kappa_{r\alpha}/\mu_\alpha$ is the phase mobility, where $\kappa_{r\alpha}$ is relative permeability and μ_α is viscosity, p_α is pressure, S_α is saturation and the well term, q_α, is flow rate per unit volume, where the subscripts α denotes the fluid phase, o or w. It is often of interest to find some parameters or control variables, v, so that some functional J is minimised, under the constraints that the equations (1) and (2) are fulfilled. To solve this type of problems, we first need state the problem on a discrete form. The reservoir is divided into a discrete number of grid cells and the time axis is divided into N intervals. The control variables are assumed to be time-dependent, so that they are given at each time step, $v = \{v^n\}_{n=1}^N$. We use a cell centred finite difference method to discretise the equations (1) and (2) which leads to a set of discrete vector equations,

$$e^n(v^n, u^n, u^{n-1}) = 0, \quad \text{for } n = 1, \dots, N, \tag{3}$$

where $u = \{u^n\}_{n=1}^N$ are the state variables,given by the pressures and saturations in each grid cell at all time steps. The state variables for time step n, u^n, are given as $u^n = \{p^n, S^n\}$. Since we have two equations for each grid cell, the length of the vector e^n is two times the number of grid cells, and furthermore the length of the state variables for time step n, u^n, is also two times the number of grid cells, since it consists of pressure and saturation in every grid cell. Thus it is possible to find a set of state variables so that equations (3) are fulfilled, if v is known. Although we assume that the control variables are time-dependent in this paper, the results in this paper does still apply if they are not dependent on time. We consider optimisation problems of the kind,

$$\min_{v,u} J(v, u), \tag{4}$$

where the objective function is of the form

$$J(v, u) = \sum_{n=1}^{N} J^n(v^n, u^n), \tag{5}$$

and an example of such a problem can be found in appendix A.

1.2 Saddle Point Problem Formulation

We can now formulate our problem as a constrained minimisation problem where we want to solve

$$\min_{v,u} J(v, u) \tag{6}$$

subject to

$$e^n(v^n, u^n, u^{n-1}) = 0, \quad \text{for } n = 1, \dots, N. \tag{7}$$

The corresponding Lagrangian is

$$L(v, u, \eta) = \sum_{n=1}^{N} L^n(v^n, u^n, u^{n-1}, \eta^n), \tag{8}$$

where

$$L^n(v^n, u^n, u^{n-1}, \eta^n) = J^n(v^n, u^n) + \eta^{nT} e^n(v^n, u^n, u^{n-1}), \tag{9}$$

and $\eta = \{\eta^n\}_{n=1}^N$ are the Lagrangian multipliers. Furthermore, we define the augmented Lagrangian by

$$L_c(v, u, \eta) = \sum_{n=1}^{N} L_c^n(v^n, u^n, u^{n-1}, \eta^n), \tag{10}$$

where

$$L_c^n(v^n, u^n, u^{n-1}, \eta^n) = L^n(v^n, u^n, u^{n-1}, \eta^n) + \frac{c}{2} e^n(v^n, u^n, u^{n-1})^T e^n(v^n, u^n, u^{n-1}), \tag{11}$$

and $c > 0$ is a penalisation constant. It is known that the solution of (7) is a saddle point of (10) for a sufficiently large penalisation parameter c, see [8] for proof. Given that the set of constraint gradients is linearly independent at the solution of (7), the following conditions hold at the solution of (7), see e.g. [21] for a proof.

Karush Kuhn Tucker (KKT) conditions Suppose that $\{v^\star, u^\star\}$ is a solution of (7). Then there exists a set of vectors $\eta^\star$ such that the following conditions hold at the point $\{v^\star, u^\star\}$,

$$\begin{aligned} \nabla_{u^n} L(v^\star, u^\star, \eta^\star) &= 0, \\ \nabla_{v^n} L(v^\star, u^\star, \eta^\star) &= 0, \\ \nabla_{\eta^n} L(v^\star, u^\star, \eta^\star) &= 0, \end{aligned} \tag{12}$$

for $n = 1, \ldots, N$. Since the last of these conditions yields $e^n = 0$, for $n = 1, \ldots, N$, it is easy to see that the KKT conditions are equivalent to

$$\begin{aligned} \nabla_{u^n} L_c(v^\star, u^\star, \eta^\star) &= 0, \\ \nabla_{v^n} L_c(v^\star, u^\star, \eta^\star) &= 0, \\ \nabla_{\eta^n} L_c(v^\star, u^\star, \eta^\star) &= 0, \end{aligned} \tag{13}$$

for $n = 1, \ldots, N$. In order to find the solution of (7), we can thus solve the KKT system (12) or equivalently (13).

2 The Augmented Lagrangian Method

The augmented Lagrangian method has previously been used to solve optimal control problems, for the case of parameter identification problems.

Itô and Kunisch [12, 13, 11] considers the augmented Lagrangian for parameter identification in elliptic systems, Chan and Tai [6] used the augmented Lagrangian method with total variations regularisation for finding discontinuous parameters in elliptic and hyperbolic systems. Chen et al. [7] also used an augmented Lagrangian approach to identify discontinuous parameters in elliptic problems.

Keung and Zou [15, 14], also considered parameter identification in parabolic and in elliptic systems, where they in the latter paper used a modified Uzawa algorithm of the augmented Lagrangian method to solve the problem.

Guo and Zou [10] also has investigated the use of the augmented Lagrangian method to identify parameters in parabolic systems.

We will use augmented Lagrangian approaches to solve the problem (6),(7), and we begin by presenting a modified Uzawa algorithm, that has been used previously to solve optimal control problems by Kunisch and Tai [16], Itô and Kunisch [12], and by Nilssen et al. [20, 19].For more information on augmented Lagrangian methods, see for example Glowinski [8, 9] and Bertsekas [2]. It is as follows,

Algorithm 1 The global optimisation algorithm

1. Choose $v_0 = \{v_0^n\}_{n=1}^N$, $\lambda_0 = \{\lambda_0^n\}_{n=1}^N$, $c > 0$. For $k = 1, 2, \ldots$ do:
2. Find $u_k = \{u_k^n\}_{n=1}^N$, such that

$$u_k = \underset{u}{\operatorname{argmin}}\, L_c(v_{k-1}, u, \lambda_{k-1}). \tag{14}$$

3. Then, find $v_k = \{v_k^n\}_{n=1}^N$ such that

$$v_k = \underset{v}{\operatorname{argmin}}\, L_c(v, u_k, \lambda_{k-1}), \tag{15}$$

4. And finally update the Lagrangian multiplier by

$$\lambda_k^n = \lambda_{k-1}^n + c\, e^n(v_k^n, u_k^n, u_k^{n-1}). \tag{16}$$

We see that the first and the second of the KKT conditions will be satisfied after the completion of (14) and (15) of the algorithm, respectively. Equation (16) of the algorithm updates the Lagrangian multipliers such that the constraints, $e^n = 0$, for $n = 1, \ldots, N$, are fulfilled at convergence. Since the two other KKT conditions are fulfilled by step 2 and 3, the solution will be found when the equation residuals $e^n = 0$. Since the number of state variables is much greater than the number of controls, the most time consuming part of this algorithm is the effort needed to solve the optimisation problem in equation (14), where the size of the problem is the number of time steps times the number of state variables, adding up to $2N \cdot M_c$ variables. This a huge number of variables, and as an attempt to make the algorithm more efficient, Nilssen and Tai proposed an improvement in [20], to be described below.

2.1 The Old Marching Scheme

As earlier mentioned, the most time consuming part in the global optimisation algorithm is the minimisation with respect to the state variables $\{u_k^n\}_{n=1}^N$. First of all,

the number of state variables is much greater than the number of controls. Furthermore since L_c^n is dependent both on u^n and on u^{n-1}, we must do the minimisation with respect all u at the same time, whereas the minimisation with respect to v can be done time step by time step since L_c^n is only dependent on v^n. In [20] and [19], Nilssen et al. tries to split the optimisation with respect to u, in equation (14) so that the optimisation for each time step is done separately using the the global optimisation algorithm with the approximation

$$u^n \approx (argmin)_{u^n} L_c^n.$$

The marching algorithm proposed by Nilssen et al. is as follows.

Algorithm 2 The Old Marching Scheme

1. Choose $v_0 = \{v_0^n\}_{n=1}^N$, $\lambda_0 = \{\lambda_0^n\}_{n=1}^N$, $c > 0$. For $k = 1, 2, \ldots$ do:
2. For $n = 1, \ldots, N$, find u_k^n such that

$$u_k^n = \underset{u^n}{\operatorname{argmin}} L_c^n(v_{k-1}^n, u^n, u_k^{n-1}, \lambda_{k-1}^n). \tag{17}$$

3. Then, for $n = 1, \ldots, N$ find v_k^n such that

$$v_k^n = \underset{v^n}{\operatorname{argmin}} L_c^n(v^n, u_k^n, u_k^{n-1}, \lambda_{k-1}^n), \tag{18}$$

4. And finally update the Lagrangian multiplier by

$$\lambda_k^n = \lambda_{k-1}^n + c\, e^n(v_k^n, u_k^n, u_k^{n-1}) \tag{19}$$

This will be done iteratively until convergence. And as in the global optimisation procedure, equation (19) of the algorithm will assure that e^n tends to zero. The minimisation in equation (18) is equivalent to the minimisation in (15), since $\frac{\partial L_c}{\partial v^n} = \frac{\partial L_c^n}{\partial v^n}$, thus the old marching scheme differs from the global optimisation algorithm in the way it calculates u in equation (17).

Contrary to the global optimisation algorithm which finds a solution satisfying the equations (13), the old marching scheme finds a solution $\{v_\dagger, u_\dagger, \lambda_\dagger\}$ satisfying the following equations, for $n = 1, \ldots, N$,

$$\begin{aligned} -\frac{\partial J^n}{\partial v^n}(v_\dagger, u_\dagger) + \lambda_\dagger^{nT} \frac{\partial e^n}{\partial v^n}(v_\dagger, u_\dagger) &= 0, \\ -\frac{\partial J^n}{\partial u^n}(v_\dagger, u_\dagger) + \lambda_\dagger^{nT} \frac{\partial e^n}{\partial u^n}(v_\dagger, u_\dagger) &= 0, \\ e^n(v_\dagger, u_\dagger) &= 0. \end{aligned} \tag{20}$$

If this is a solution of (7), the solution must satisfy the equations (13). Thus a solution found by the old marching scheme can only be a solution of (7) if

$$\frac{\partial L_c^m}{\partial u^n}(v_\star, u_\star, \lambda_\star) = 0, \quad \text{for} \quad \mathrm{m} \neq \mathrm{n} \quad \text{and} \quad \mathrm{n}, \mathrm{m} \in \{1, \ldots, \mathrm{N}\}, \tag{21}$$

and if

$$\frac{\partial L_c^m}{\partial v^n}(v_\star, u_\star, \lambda_\star) = 0, \quad \text{for} \quad \mathrm{m} \neq \mathrm{n} \quad \text{and} \quad \mathrm{n}, \mathrm{m} \in \{1, \ldots, \mathrm{N}\}. \tag{22}$$

From our model we have that, for $n = 1, \ldots, N-1$

$$\frac{\partial L_c}{\partial u^n} = \frac{\partial L_c^n}{\partial u^n} + \frac{\partial L_c^{n+1}}{\partial u^n} \tag{23}$$

$$= \frac{\partial \boldsymbol{J}^n}{\partial u^n} + \frac{\partial e^n}{\partial u^n}^T (\lambda^n + c \cdot e^n) + \frac{\partial e^{n+1}}{\partial u^n}^T (\lambda^{n+1} + c \cdot e^{n+1}), \tag{24}$$

and that

$$\frac{\partial L_c}{\partial u^N} = \frac{\partial L_c^N}{\partial u^N} = \frac{\partial \boldsymbol{J}^N}{\partial u^N} + \frac{\partial e^N}{\partial u^N}^T (\lambda^N + c \cdot e^N). \tag{25}$$

Now, for $n = 1, \ldots, N$, if $\frac{\partial J^n}{\partial u^n}(v_\star^n, u_\star^n) = 0$, we see that must have that $\lambda_\star^n = 0$ since $e^n(v_\star^n, u_\star^n, u_\star^{n-1})$, showing that the old marching scheme finds a solution to the problem if we have that

$$\frac{\partial \boldsymbol{J}^n}{\partial u^n}(v_\star^n, u_\star^n) = 0, \quad \text{for} \quad n = 1, \ldots, N. \tag{26}$$

Condition (26) will typically be fulfilled if we have a least-squares objective function where one expects that the objective function will be close to zero at the minimum. For the problems that was solved by Nilssen in [19] and in [18] this has been the case. But for problems, with an objective function which is not close to zero at the optimum this marching scheme cannot find the optimum.

2.2 A New Marching Scheme

In this paper, we try to find a way to circumvent the problem of the old marching scheme, and keeping the idea of doing the minimisation in u sequentially. In the old marching scheme the optimisation is done sequentially by neglecting the gradient term

$$\frac{\partial L_c^{n+1}}{\partial u^n} = \left(\lambda^{n+1} + c \cdot e^{n+1}\right)^T \frac{\partial e^{n+1}}{\partial u^n}, \tag{27}$$

to improve the speed of the global optimisation algorithm. This may work, when condition (21) is fulfilled, but since this is not the case for our problem we cannot use the old marching scheme.

Here we present a different way to approximate the minimisation with respect to u in the global optimisation algorithm, such that this sub-minimisation problem can be solved sequentially. We want to do the minimisation with respect to u sequentially, time step by time step, to improve the speed of the algorithm. Since e^{n+1} depends on u^{n+1}, which is unknown at iteration n, we must do some sort of approximation. When this minimisation is done in the old marching scheme, we find u^n such that

$$-\frac{\partial \boldsymbol{J}^n}{\partial u^n} + \left(\lambda^n + c \cdot e^n\right)^T \frac{\partial e^n}{\partial u^n} = 0.$$

But it is not necessary to neglect the term (27) entirely, because it is only e^{n+1} which depends on u^{n+1}. We propose instead to find u^n sequentially such that

$$-\frac{\partial J^n}{\partial u^n} + \left(\lambda^n + c \cdot e^n\right)^T \frac{\partial e^n}{\partial u^n} + {\lambda^{n+1}}^T \frac{\partial e^{n+1}}{\partial u^n} = 0. \tag{28}$$

From appendix B we have that

$$\frac{\partial e^n}{\partial u^{n-1}} = \begin{pmatrix} \frac{\partial e_1^n}{\partial p_1^{n-1}} & \cdots & \frac{\partial e_1^n}{\partial p_m^{n-1}} & \frac{\partial e_1^n}{\partial s_1^{n-1}} & \cdots & \frac{\partial e_1^n}{\partial s_m^{n-1}} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e_m^n}{\partial p_1^{n-1}} & \cdots & \frac{\partial e_m^n}{\partial p_m^{n-1}} & \frac{\partial e_m^n}{\partial s_1^{n-1}} & \cdots & \frac{\partial e_m^n}{\partial s_m^{n-1}} \\ \frac{\partial e_{m+1}^n}{\partial p_1^{n-1}} & \cdots & \frac{\partial e_{m+1}^n}{\partial p_m^{n-1}} & \frac{\partial e_{m+1}^n}{\partial s_1^{n-1}} & \cdots & \frac{\partial e_{m+1}^n}{\partial s_m^{n-1}} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e_{2m}^n}{\partial p_1^{n-1}} & \cdots & \frac{\partial e_{2m}^n}{\partial p_m^{n-1}} & \frac{\partial e_{2m}^n}{\partial s_1^{n-1}} & \cdots & \frac{\partial e_{2m}^n}{\partial s_m^{n-1}} \end{pmatrix}, \tag{29}$$

where

$$\frac{\partial e_i^n}{\partial s_j^{n-1}} = \begin{cases} \phi_j & \text{if } i = j \\ -\phi_j & \text{if } i = j + m \\ 0 & \text{else} \end{cases} \tag{30}$$

This leads to

$$\frac{\partial e^n}{\partial u^{n-1}} = \begin{pmatrix} 0 & \dots & \dots & 0 & \phi_1 & 0 & \dots & 0 \\ \vdots & \ddots & & \vdots & 0 & \phi_2 & & \vdots \\ \vdots & & \ddots & \vdots & \vdots & & \ddots & 0 \\ 0 & \dots & \dots & 0 & 0 & \dots & 0 & \phi_m \\ 0 & \dots & \dots & 0 & -\phi_1 & 0 & \dots & 0 \\ \vdots & \ddots & & \vdots & 0 & -\phi_2 & & \vdots \\ \vdots & & \ddots & \vdots & \vdots & & \ddots & 0 \\ 0 & \dots & \dots & 0 & 0 & \dots & 0 & -\phi_m \end{pmatrix}, \tag{31}$$

since $\frac{\partial e_i^n}{\partial p_j^{n-1}} = 0, \forall i, j$. Furthermore we observe that finding u^n as

$$u^n = \underset{u^n}{\operatorname{argmin}} \left[-J^n + \left(\lambda^n + \frac{c}{2} e^n\right)^T e^n + {\lambda^{n+1}}^T \tilde{S}^{n+1} \right], \tag{32}$$

where $\tilde{S}^{n+1}$ is given by

$$\tilde{S}^{n+1} = [\phi_1 s_1^n, \dots, \phi_m s_m^n, -\phi_1 s_1^n, \dots, -\phi_m s_m^n]^T, \tag{33}$$

is equivalent to finding u^n such that (28) is satisfied. In fact, if e^n approaches 0 for $n = 1, \dots, N$ at convergence, this algorithm will satisfy the KKT conditions of (7) and thus solve the problem. Let us define $\tilde{L}_c^n$ by

$$\tilde{L_c^n} = -J^n + \left(\lambda^n + \frac{c}{2} e^n\right)^T e^n + {\lambda^{n+1}}^T \tilde{S}^{n+1}. \tag{34}$$

Now the new marching algorithm is as follows.

Algorithm 3 New Marching Scheme

1. Choose $v_0 = \{v_0^n\}_{n=1}^N$, $\lambda_0 = \{\lambda_0^n\}_{n=1}^N$, $c > 0$. For $k = 1, 2, \ldots$ do:
2. For $n = 1, \ldots, N$, find u_k^n such that

$$u_k^n = \underset{u^n}{\operatorname{argmin}} \tilde{L}_c^n(v_{k-1}^n, u^n, u_k^{n-1}, \lambda_{k-1}^{n+1}, \lambda_{k-1}^n). \tag{35}$$

3. Then, for $n = 1, \ldots, N$ find v_k^n such that

$$v_k^n = \underset{v^n}{\operatorname{argmin}} L_c^n(v^n, u_k^n, u_k^{n-1}, \lambda_{k-1}^n), \tag{36}$$

4. And finally update the Lagrangian multiplier by

$$\lambda_k^n = \lambda_{k-1}^n + c\, e^n(v_k^n, u_k^n, u_k^{n-1}). \tag{37}$$

This will be done iteratively until convergence. And as in the previous optimisation algorithms, equation (37) of the algorithm will assure that e^n, for $n = 1, \ldots, N$, tends to zero. For the minimisation in step (35) and (36) we use the LBFGS method, where we use projected gradients to handle the bounds on the variables. For more information regarding the LBFGS method see Byrd et al. [5] and [4].

3 Gauss-Seidel scheme

Another way of solving the problem of the Old marching scheme is to include the term (27), by simply using the state variables from the previous iteration to calculate the e^{n+1}. Thus we will do the optimisation sequentially, time step by time step, always using the most recently calculated variables available. This has been done before by Nilssen at al. in [19] and [18]. In the minimising concerning u^n, we will thus find u_k^n such that

$$u_k^n = \underset{u^n}{\operatorname{argmin}} L_c(v_{k-1}^1, \ldots, v_{k-1}^N, u_k^1, \ldots, u_k^{n-1}, u^n, u_{k-1}^1, \ldots, u_{k-1}^{n-1}, \lambda_{k-1}^1, \ldots, \lambda_{k-1}^N), \tag{38}$$

that is

$$u_k^n = \underset{u^n}{\operatorname{argmin}} \left[L_c^n(v_{k-1}^n, u_k^{n-1}, u^n, \lambda_{k-1}^n) + L_c^{n+1}(v_{k-1}^{n+1}, u^n, u_{k-1}^{n+1}, \lambda_{k-1}^{n+1}) \right]. \tag{39}$$

This leads to the algorithm

Algorithm 4 Gauss-Seidel Scheme

1. Choose $v_0 = \{v_0^n\}_{n=1}^N$, $\lambda_0 = \{\lambda_0^n\}_{n=1}^N$, $c > 0$. For $k = 1, 2, \ldots$ do:
2. For $n = 1, \ldots, N$, find u_k^n such that

$$u_k^n = \underset{u^n}{\operatorname{argmin}} \left[L_c^n(v_{k-1}^n, u_k^{n-1}, u^n, \lambda_{k-1}^n) + L_c^{n+1}(v_{k-1}^{n+1}, u^n, u_{k-1}^{n+1}, \lambda_{k-1}^{n+1}) \right] \tag{40}$$

3. Then, for $n = 1, \ldots, N$ find v_k^n such that

$$v_k^n = \underset{v^n}{\operatorname{argmin}}\, L_c^n(v^n, u_k^n, u_k^{n-1}, \lambda_{k-1}^n), \tag{41}$$

4. And finally update the Lagrangian multiplier by

$$\lambda_k^n = \lambda_{k-1}^n + c\, e^n(v_k^n, u_k^n, u_k^{n-1}). \tag{42}$$

This will be done iteratively until convergence, which will occur when e^n, for $n = 1, \ldots, N$ approaches zero. For the minimisation in step (40) and (2) we use, as for the new marching scheme, the LBFGS method where projected gradients are used to handle the bounds on the variables.

3.1 Comparison With The Adjoint Method

To compare the results of the new marching scheme and the Gauss-Seidel scheme, we use an optimal control theory method, the adjoint method, used previously in [3], [17], [22] and [23], to solve the production optimisation problem. Letting, as for the earlier algorithms, the subscript k be the outer iteration counter, the adjoint algorithm is as follows.

The Adjoint Algorithm

1. Choose $\hat{v}_0 = \{\hat{v}_0^n\}_{n=1}^N$, $\hat{u}_0^0$ and $c > 0$. For $k = 1, 2, \ldots$ do:
2. For $n = 1, 2, \ldots, N$, find $\hat{u}_k^n$ such that

$$e^n(\hat{v}_{k-1}^n, \hat{u}_k^n, \hat{u}_k^{n-1}) = 0. \tag{43}$$

3. Find η_k^N such that

$$-\frac{\partial \boldsymbol{J}^N}{\partial \hat{u}^N}(\hat{v}_{k-1}^N, \hat{u}_k^N) + \eta_k^{N\,T} \frac{\partial e^N}{\partial \hat{u}^N}(\hat{v}_{k-1}^N, \hat{u}_k^N, \hat{u}_k^{N-1}) = 0. \tag{44}$$

4. For $n = N-1, N-2, \ldots, 1$, find η_k^n, such that

$$-\frac{\partial \boldsymbol{J}^n}{\partial \hat{u}^n}(\hat{v}_{k-1}^n, \hat{u}_k^n) + \eta_k^{n\,T} \frac{\partial e^n}{\partial \hat{u}^n}(\hat{v}_{k-1}^n, \hat{u}_k^n, \hat{u}_k^{n-1}) + \eta_k^{n+1\,T} \frac{\partial e^{n+1}}{\partial \hat{u}^n}(\hat{v}_{k-1}^{n+1}, \hat{u}_k^{n+1}, \hat{u}_k^n) = 0. \tag{45}$$

5. Finally we find the gradient of $-\boldsymbol{J}$ as

$$-\frac{d\boldsymbol{J}^n}{d\hat{v}^n}(\hat{v}^n_{k-1}) = \frac{\partial \boldsymbol{L}_c}{\partial \hat{v}^n}(\hat{v}^n_{k-1}, \hat{u}^n_k) \tag{46}$$
$$= -\frac{\partial \boldsymbol{J}^n}{\partial \hat{v}^n}(\hat{v}^n_{k-1}, \hat{u}^n_k) + \eta^{nT}_k \frac{\partial e^n}{\partial \hat{v}^n}(\hat{v}^n_{k-1}, \hat{u}^n_k, \hat{u}^{n-1}_k) = 0,$$

and use this gradient in a the LBFGS algorithm to find $\hat{v}_k = \{\hat{v}^n_k\}_{n=1}^N$.

Equation (43) in the algorithm is in fact the same as solving the forward problem, and when solving the non-linear systems of equations in (43), we use Newton's method so that we find $\hat{u}^n_k$ as

- Choose $\hat{u}^n_{k,0}$. For $l = 1, 2, \ldots$ do until convergence: For $n = 1, 2, \ldots, N$ do:

$$\hat{u}^n_{k,l} = \hat{u}^n_{k,l-1} - \left(\frac{\partial e^n}{\partial \hat{u}^n}(\hat{v}^n_{k-1}, \hat{u}^n_{k,l-1})\right)^{-1} e^n(\hat{v}^n_{k-1}, \hat{u}^n_{k,l-1}, \hat{u}^{n-1}_{k,l}), \tag{47}$$

where we solve the linear system of equations with the GMRES method.

In order to find η^n for the adjoint algorithm we need to solve a linear system of equations in (44) and in (45), for which we use the GMRES method.

4 Numerical Examples

In order to test the methods presented so far, we study the production optimisation problem, presented in appendix A, and how this problem is discretised numerically is presented in appendix B, with corresponding derivatives in appendix C. Here we present two different examples. We use a synthetic reservoir, with dimensions $450m \times 450m \times 10m$, which is depicted in figure 1. There is a high permeable channel going from the side of the side of the reservoir where the injector is located to the side of the reservoir where the producer is located. We use a 5×5 grid, and let there be one valve opening in each grid cell along the injector and the producer. We will inject one pore volume over a period of two years, dividing this time interval into 64 time steps. As an initial guess for our rates, we use uniform injection and production, such that the individual injection and production rates are equal for all the valve openings. Furthermore, the initial value of the Lagrangian multipliers is $\lambda^n_0 = 0$ for $n = 1, \ldots, N$, for both methods. Before starting to produce, the water saturation is the residual water saturation of 0.2 everywhere in the reservoir, and the initial pressure is 190 bar, i.e. $19 \cdot 10^6$ Pa, in the entire reservoir. Other reservoir simulator constants are given in table 1.

For the revenue of oil produced, we set $I_o = 80\ \$/m^3$ and set the cost associated with water production to $I_w = -20\ \$/m^3$. Although the revenue constant is very low comparing with the prices of today, it is set deliberately to this value since this has been used in earlier studies in [3]. The annual interest rate is set to 20%, in equation (64). Since the constraints are only fulfilled at convergence for the

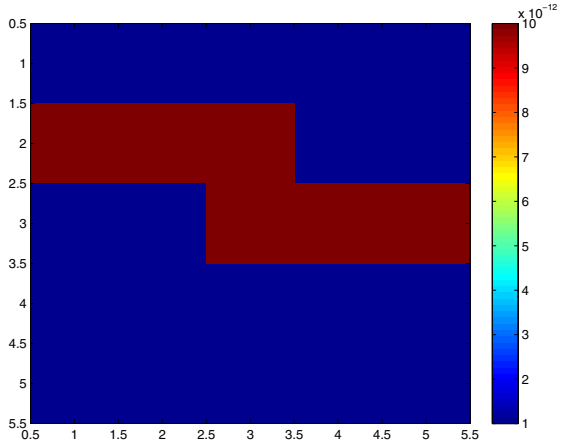

Figure 1: Permeability field 1, given in m^2.

κ_{ro}	κ_{rw}	e_o	e_w	S_{or}	S_{wr}	μ_o	μ_w	ϕ
1.0	0.1	2.0	1.5	0.2	0.2	0.5	0.5	0.2

Table 1: Reservoir simulator constants -equal for all examples.

augmented Lagrangian methods, it is not obvious to see how good our current estimate of the controls v_k is. Thus as an illustration of what the profit will be, if we terminate the algorithm at iteration k we solve the forward problem using the latest set of control variables. The state variables when solving the forward problem given the rates v_k will be denoted $\tilde{u}_k$. That is $\tilde{u}_k = \{u|e^n(v_k^n, u^n, u^{n+1}) = 0,$ for n $= 1, \ldots,$ N$\}$. Correspondingly, $\tilde{J}_k = J(v_k, \tilde{u}_k)$ as opposed to $J_k = J(v_k, u_k)$ which is the objective function calculated from a current iteration of one of the augmented Lagrangian methods. For the different examples, we also plot $J_k = J(v_k, u_k)$ and $\tilde{J}_k$ versus the number of iterations of the different algorithms and we also provide plots of the development of $\sqrt{\sum_{n=1}^{N} ||e^n(v_k, u_k)||^2/(2 \cdot M_c N)}$ versus iterations of the KKT algorithm, to show the convergence of the residual for the examples.

4.1 Example 1, Two-Parameter Example

To test the old and new marching scheme and the Gauss-Seidel scheme, it is of much help to test on a problem where the solution is known. Therefore we will construct an example with two controls, so that it is possible to plot the solution as a function of the two controls. We let the production be uniformly distributed amongst the producer's valve openings, so that $v_6^n = v_7^n = v_8^n = v_9^n = v_{10}^n = 0.2$, for $n = 1, \ldots, 64$. Furthermore, we let the injection rates in the four uppermost valve openings equal each otherso that $v_2^n = v_3^n = v_4^n = v_5^n$, for $n = 1, \ldots, 64$. Obviously,

c	10^7	10^8	10^9	Adjoint
NPV	$1.5525 \cdot 10^7$	$1.5525 \cdot 10^7$	$1.5524 \cdot 10^7$	$1.5526 \cdot 10^7$
$\bar{v}_1$	0.2049	0.1921	0.2748	0.2422
$\bar{v}_2$	0.5969	0.5399	0.3934	0.5011

Table 2: For example 1, the final NPV found with the new marching scheme.

c	10^7	10^8	10^9	Adjoint
NPV	$1.5233 \cdot 10^7$	$1.5514 \cdot 10^7$	$1.5521 \cdot 10^7$	$1.5526 \cdot 10^7$
$\bar{v}_1$	1.0000	1.0000	0.2180	0.2422
$\bar{v}_2$	0.2000	0.0000	0.4417	0.5011

Table 3: For example 1, the final NPV found with the Gauss-Seidel scheme.

since $\sum_{i=1}^{5} v_i^n = 1$, we have that $v_1^n = (1 - v_j^n)/4$, where $j \in \{2,3,4,5\}$, so that by knowing v_1^n, we automatically know v_2^n,v_3^n, v_3^n and v_5^n. Additionally we will only have the possibility to change the injection rates, once after time step number 20, so that we are left with only two controls, v_1^n for $n = 1,\ldots,20$ and v_1^n for $n = 21,\ldots,64$. We will denote $\bar{v}_1 = v_1^1 = \ldots = v_1^{20}$ and $\bar{v}_2 = v_1^{21} = \ldots = v_1^{64}$. Since we now have an optimisation problem in two variables, it is easy to plot the solution, and this is done in figure 2, where $\bar{v}_1$ and $\bar{v}_2$ are denoted as rate 1 and rate 2 respectively. We have tested the new marching scheme for three different values of the penalty parameter c, and in figure 3 we plot the development of the $\tilde{J}$, as a function of iterations for the new marching scheme. In table 2, the maximum NPV for the three different values of c and for the adjoint method is shown. We see from this table that we obtain approximately the same NPV for the three different choices of c and for the adjoint method. Furthermore, we see that the values for $\bar{v}_1$ and $\bar{v}_2$ are close to the maximum region, that can be seen in figure 2, thus all giving satisfactorily NPV values. However, when studying figure 3 we see that $\tilde{J}$ is oscillating for the choices $c = 10^7$ and $c = 10^8$, and when we set $c = 10^9$, the NPV is monotonically increasing. Since the choice of $c = 10^9$ seems to be the best, we show in figure 4 the plot of J versus iterations for the choice of $c = 10^9$ and we plot the mean residual $\sqrt{\sum_{n=1}^{N} ||e^n(v_k, u_k)||^2/(2 \cdot M_c N)}$ in figure 5.

Furthermore we have also tested the Gauss-Seidel scheme with three different values of the penalty constant c, and the development of $\tilde{J}$ for the three different c-values as a function of iterations is shown in figure 6. The resulting NPV, and the

c	10^7	10^8	10^9	Adjoint
NPV	$1.5505 \cdot 10^7$	$1.5509 \cdot 10^7$	$1.5477 \cdot 10^7$	$1.5526 \cdot 10^7$
$\bar{v}_1$	0.1912	0.1986	0.1998	0.2422
$\bar{v}_2$	0.3902	0.7581	0.2853	0.5011

Table 4: For example 1, the final NPV found with the old marching scheme.

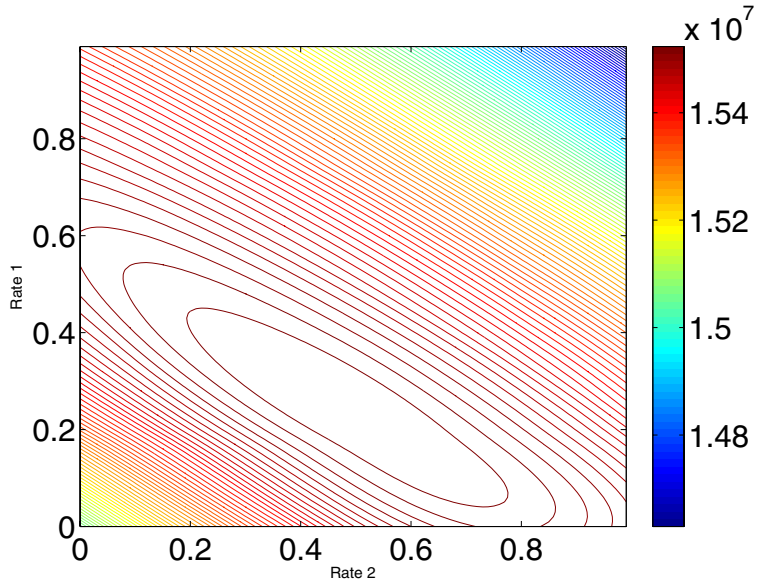

Figure 2: Profit function for example 1.

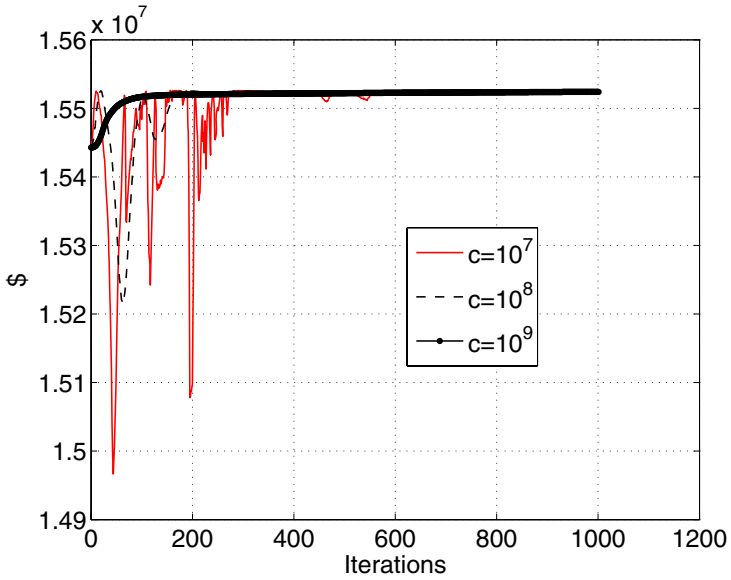

Figure 3: Example 1. The development of the NPV as a function of iterations for the new marching scheme, with three different values of the penalty parameter c.

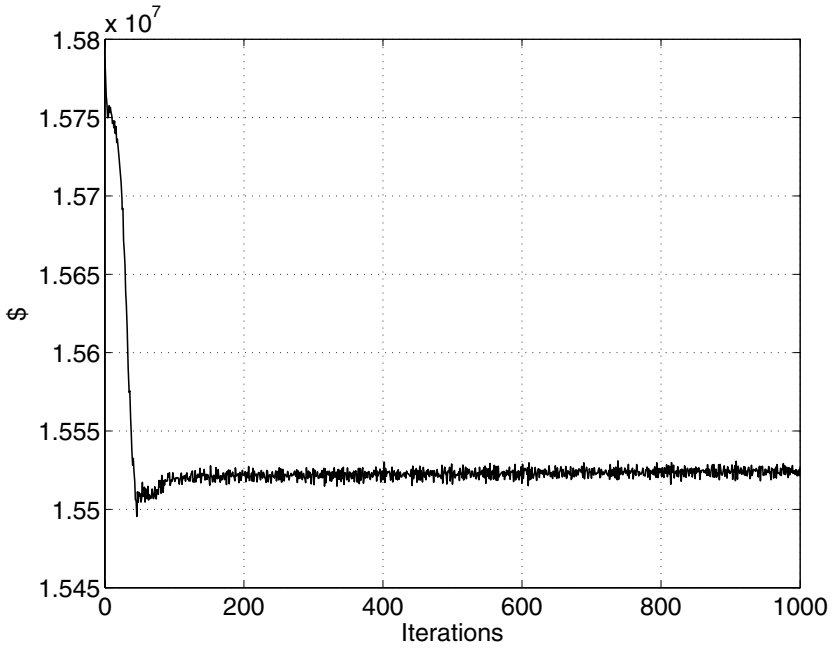

Figure 4: Example 1, the new marching scheme. Profit for the choice of $c = 10^9$

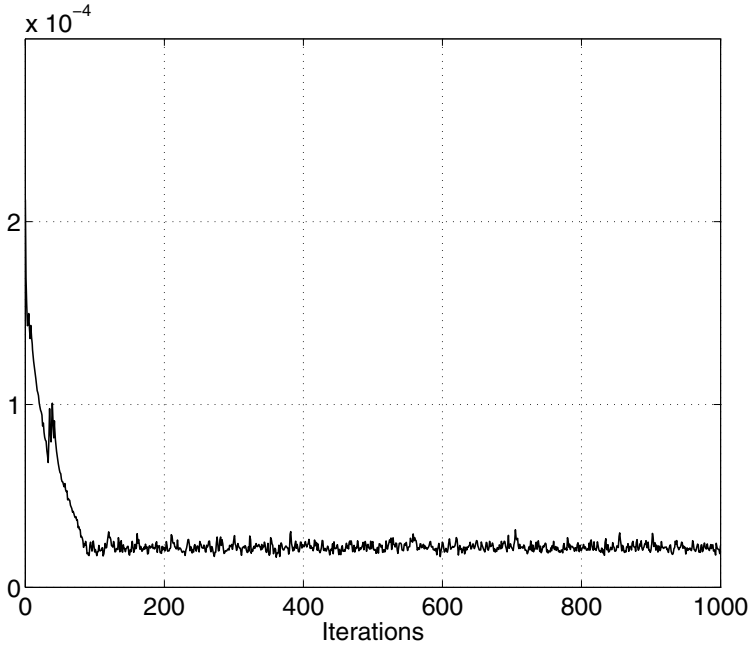

Figure 5: Example 1, the new marching scheme. Residual for the choice of $c = 10^9$

rates found are stated in table 3. We see from figure 6 that the choices of $c = 10^7$ and $c = 10^8$ leads to a highly oscillatory $\tilde{J}$. Furthermore, the resulting rates for these two c-values, are far from the maximum of the the NPV function shown in figure 2. For $c = 10^9$ the Gauss-Seidel scheme, we see that $\tilde{J}$ is not oscillating as much as the other two. Additionally, we see that the optimal NPV found, is close to the NPV found by the adjoint method, and we also see that the optimal rates are close to the maximum of the NPV function. In figure 7 we show the development of J, and in figure 8 we plot the mean residual $\sqrt{\sum_{n=1}^{N} ||e^n(v_k, u_k)||^2/(2 \cdot M_c N)}$, both for case when $c = 10^9$.

We have also tried to solve the problem with the old marching scheme, with three different values of c. In figure 9, $\tilde{J}$ is shown as a function of iterations for this scheme, and the resulting NPV can be found in table 4. From table 4, we see that the choices of $c = 10^8$ and $c = 10^7$ results in approximately the same NPV, with rates that are close to the maximum of the function which is depicted in figure 2. The choice of $c = 10^9$ does not give such a good NPV. From figure 9, we see that the choice of $c = 10^8$ gives a highly oscillatory $\tilde{J}$, while the curve of $\tilde{J}$ for $c = 10^7$ is not oscillating and seems thus to be the better. In figure 10 we show the development of J for the old marching scheme for the choice of $c = 10^7$ and in figure 11 we show the mean residual $\sqrt{\sum_{n=1}^{N} ||e^n(v_k, u_k)||^2/(2 \cdot M_c N)}$. From figure 8, we see that the residual does not converge to zero, thus the method does not seem to be converging for the choice of $c = 10^7$.

4.2 Example 2

In this example we have the possibility to control the injection and the production in all well segments along the injector and the producer at all time steps. For our 5×5 grid, we have five valve openings in the injector and five in the producer - one in each grid cell along each side of the reservoir, and we can change the rates in these valve openings at each of our 64 time steps. We solve the problem with the adjoint method, and the resulting rates can be found in figure 24, where the injection rates are in the top sub-figure and the production rates are in the bottom sub-figure. Along the horizontal axis, we have the time steps and we along the vertical axis we have the different valve openings.

The new marching scheme is tested for three different choices of the penalty parameter c, and we have plotted $\tilde{J}$ against the number of iterations for these three choices in figure 12. In table 5 we can see the resulting maximum profit after 1000 iterations for the new marching scheme and for the adjoint method. All the three choices of c gives an NPV that is about the same as for the adjoint method, but the one corresponding to $c = 10^8$ finds the highest NPV of the three. When investigating figure 12, we see that the NPV is oscillating when $c = 10^7$ and $c = 10^8$, and that the curve of $\tilde{J}$ is monotonically increasing. This shows that the choice of $c = 10^8$ finds higher NPV than the choice of $c = 10^9$, but the latter seems to be more stable. We show the plot of J in figure 13 and the plot of $\sqrt{\sum_{n=1}^{N} ||e^n(v_k, u_k)||^2/(2 \cdot M_c N)}$

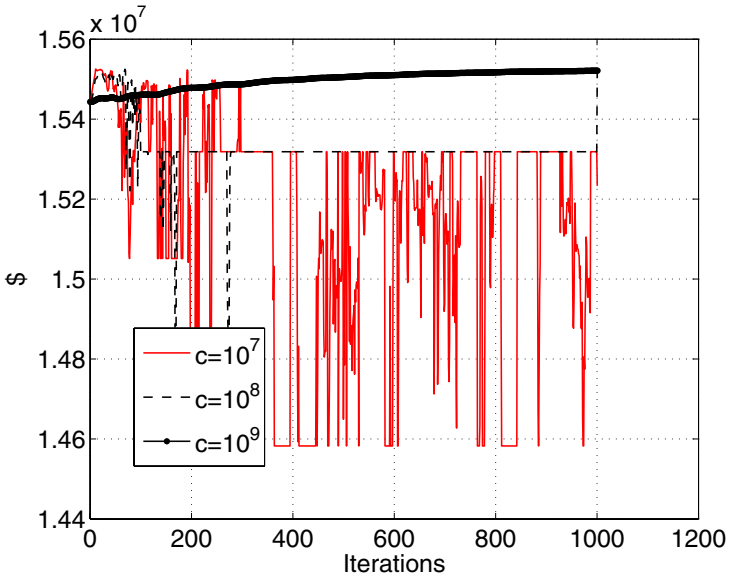

Figure 6: Example 1. The development of the NPV as a function of iterations for the new Gauss-Seidel, with three different values of the penalty parameter c.

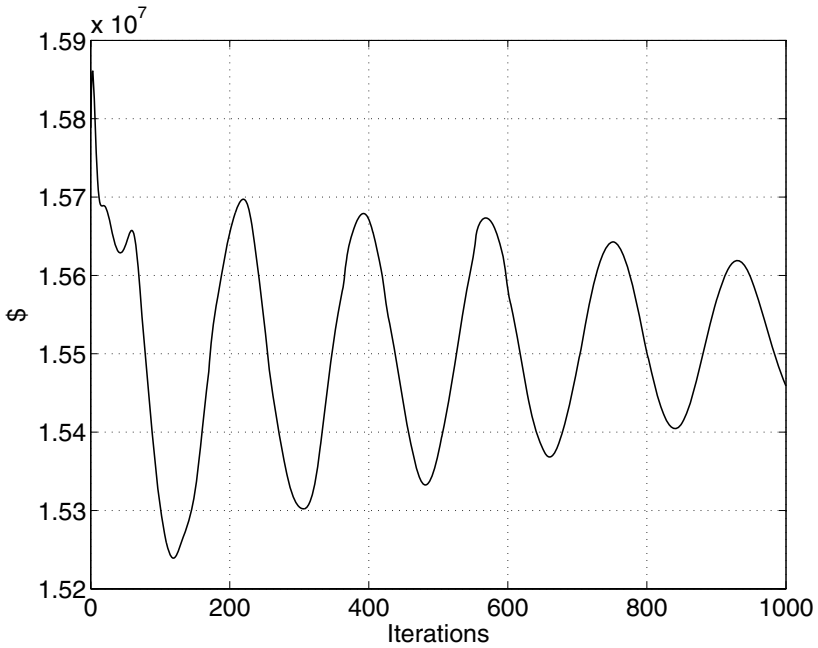

Figure 7: Example 1, the Gauss-Seidel scheme. Profit for the choice of $c = 10^9$

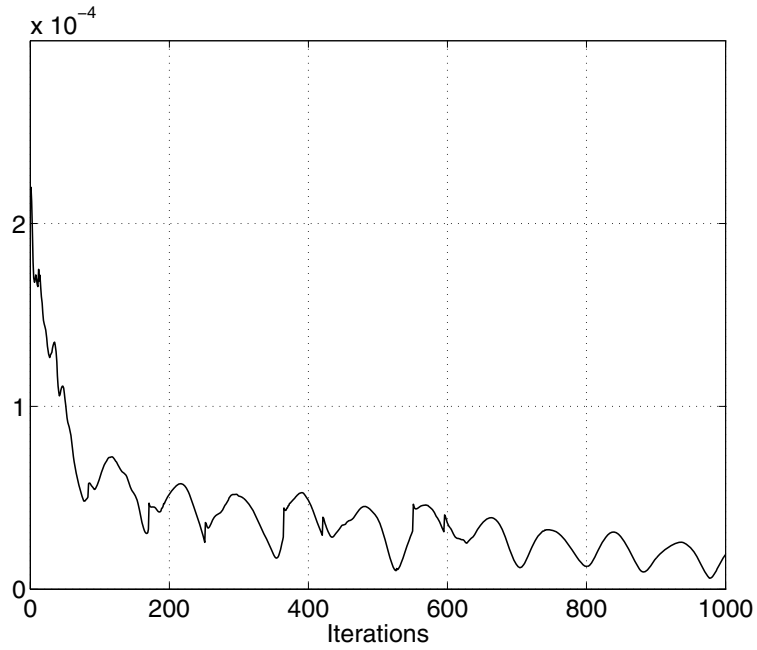

Figure 8: Example 1, the Gauss-Seidel scheme. Residual for the choice of $c = 10^9$

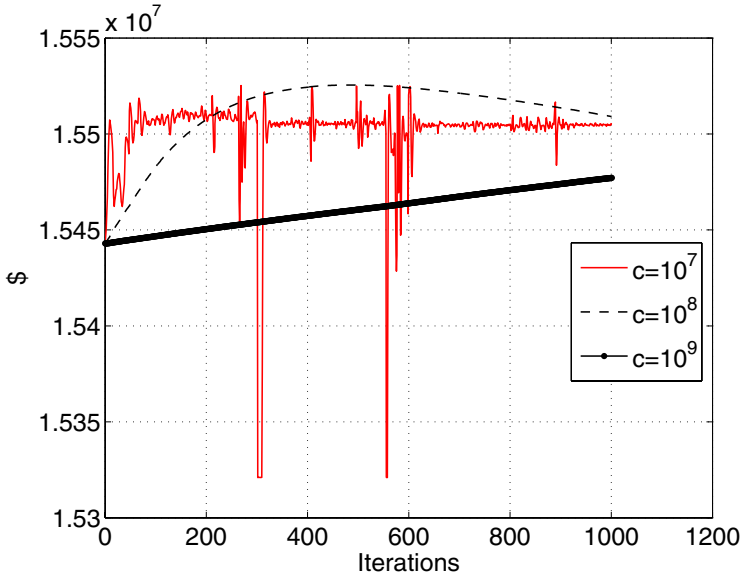

Figure 9: Example 1. The development of the NPV as a function of iterations for the old marching scheme, with three different values of the penalty parameter c.

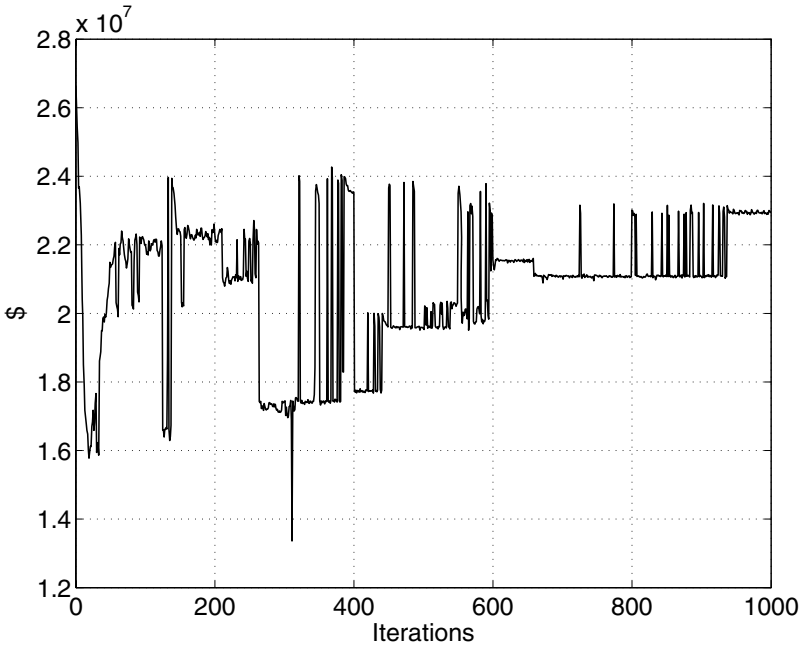

Figure 10: Example 1, the old marching scheme. Profit for the choice of $c = 10^7$

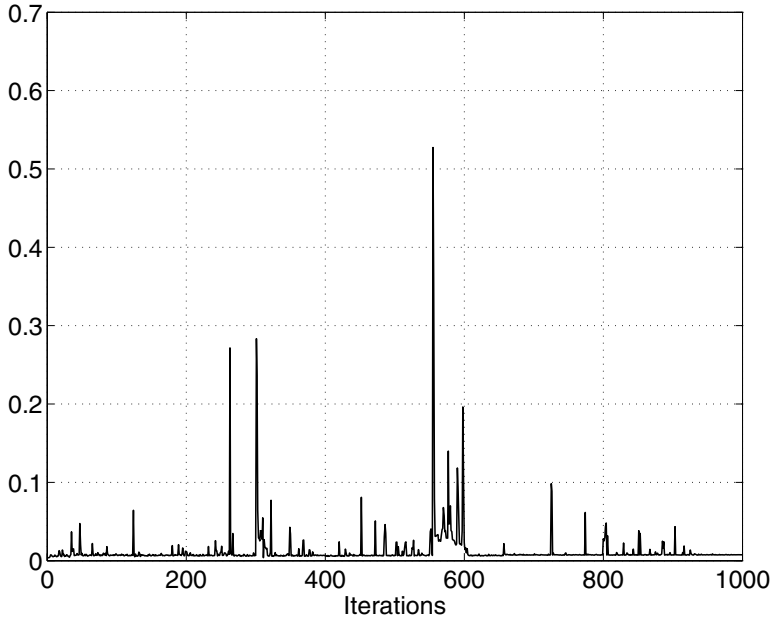

Figure 11: Example 1, the old marching scheme. Residual for the choice of $c = 10^7$

c	10^7	10^8	10^9	Adjoint
NPV	$1.6018 \cdot 10^7$	$1.6043 \cdot 10^7$	$1.6026 \cdot 10^7$	$1.6048 \cdot 10^7$

Table 5: For example 2, the final NPV found with the new marching scheme.

c	10^7	10^8	10^9	Adjoint
NPV	$1.5443 \cdot 10^7$	$1.5444 \cdot 10^7$	$1.5997 \cdot 10^7$	$1.6048 \cdot 10^7$

Table 6: For example 2, the final NPV found with the Gauss-Seidel scheme.

in figure 14 to show how the method proceeds during the iterations. The final rates found after 1000 iterations with the new marching scheme, for $c = 10^9$ are shown in figure 15, where the injection rates are in the top sub-figure and the production rates are in the bottom sub-figure. Along the horizontal axis, we have the time steps and we along the vertical axis we have the different valve openings.

For the Gauss-Seidel scheme, we have also done the optimisation with three different values of the penalty constant c, and the resulting $\tilde{J}$ is plotted versus iterations in figure 16. In table 6, the resulting maximum profit after 1000 iterations is shown. From this table, we see that the Gauss-Seidel scheme does not find as high NPV as the adjoint method. Since the choice of $c = 10^9$ finds the highest NPV, we have plotted the development of J in figure 17 and $\sqrt{\sum_{n=1}^{N} ||e^n(v_k, u_k)||^2/(2 \cdot M_c N)}$ in figure 18, to illustrate the behaviour of the method. Also for this method, we plot the resulting rates, in figure 19.

The development of $\tilde{J}$, for the old marching scheme is plotted in figure 20, where we used the same three different values of c as for the two other methods. From this figure, we see that the choice of $c = 10^7$ is very oscillating, but the two other choices of c seems to be more stable. In table 7, the resulting maximum profit after 1000 iterations is shown, and we see that the choice of $c = 10^8$ results in the highest NPV, but it is not as high as the NPV found with the adjoint method. For the case of $c = 10^8$, we plot the development of J in figure 21 and $\sqrt{\sum_{n=1}^{N} ||e^n(v_k, u_k)||^2/(2 \cdot M_c N)}$ in figure 22. The final rates found, are shown in figure 23.

5 Conclusions

In this paper we investigate use different marching schemes of the augmented Lagrangian method and apply them to an optimal control problem that occurs in reservoir engineering. We have shown the shortcomings of the old marching scheme,

c	10^7	10^8	10^9	Adjoint
NPV	$1.5670 \cdot 10^7$	$1.5950 \cdot 10^7$	$1.5946 \cdot 10^7$	$1.6048 \cdot 10^7$

Table 7: For example 2, the final NPV found with the old marching scheme.

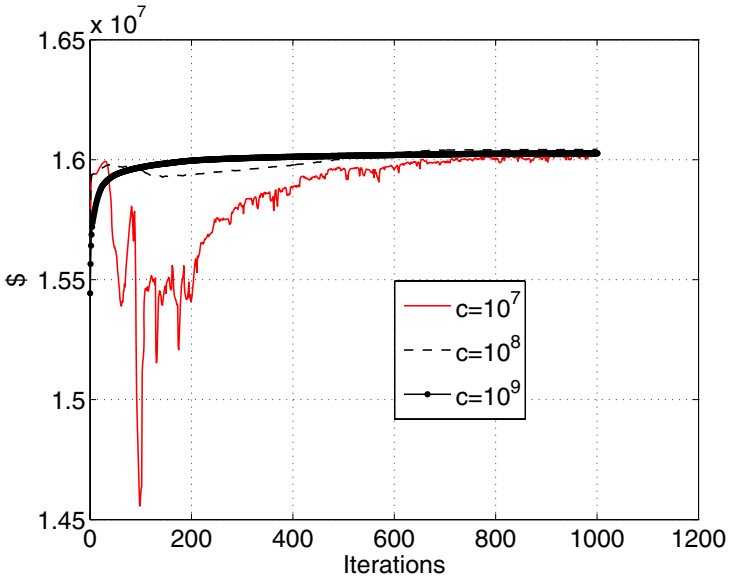

Figure 12: Example 2: Comparisons of the profit obtained with the new marching scheme, with three different values of c.

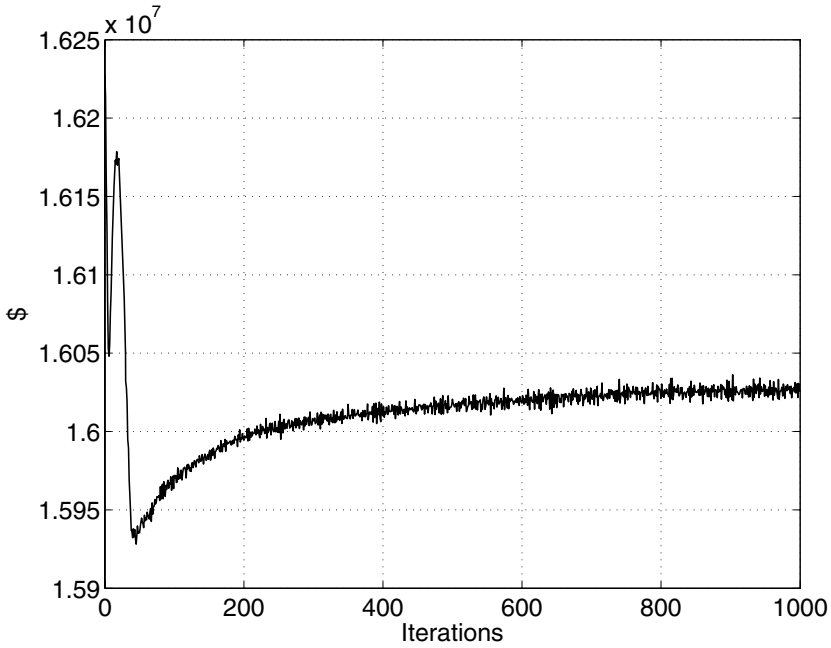

Figure 13: Example 2, the new marching scheme. Profit for the choice of $c = 10^9$

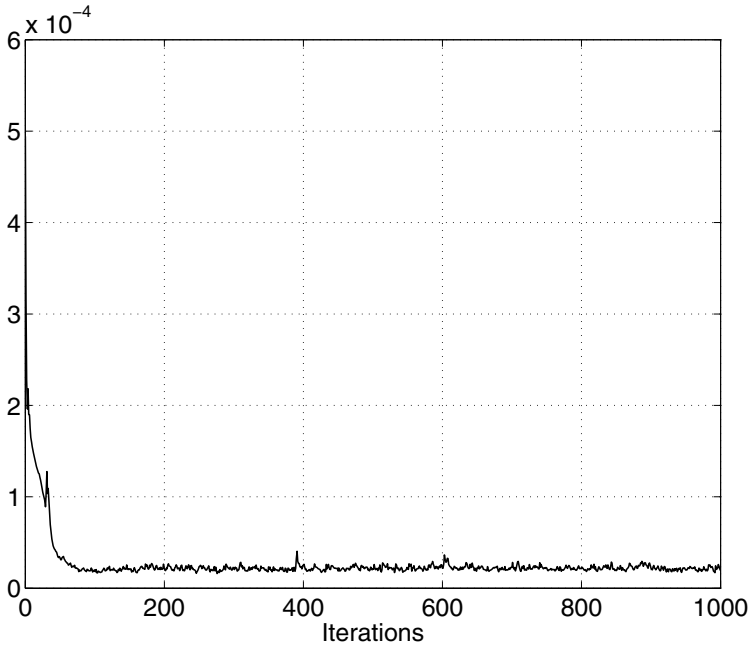

Figure 14: Example 2, the new marching scheme. Residual for the choice of $c = 10^9$

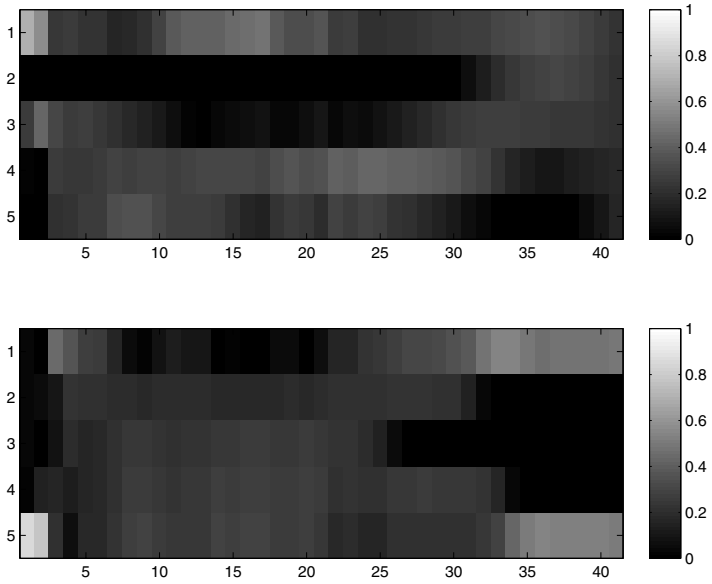

Figure 15: Example 2, the new marching scheme. Optimal rates found for the choice of $c = 10^9$

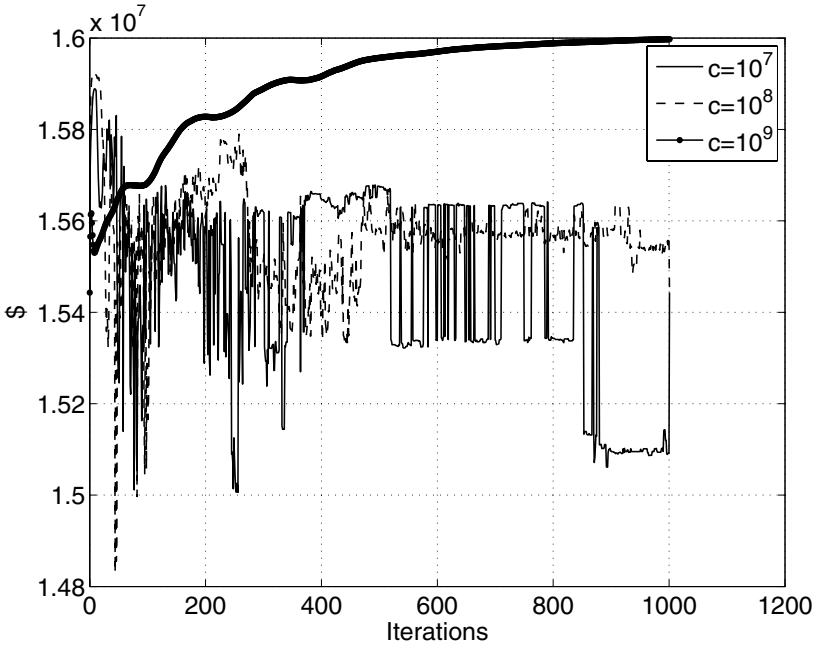

Figure 16: Example 2: Comparisons of the profit obtained with the Gauss-Seidel scheme, with three different values of c.

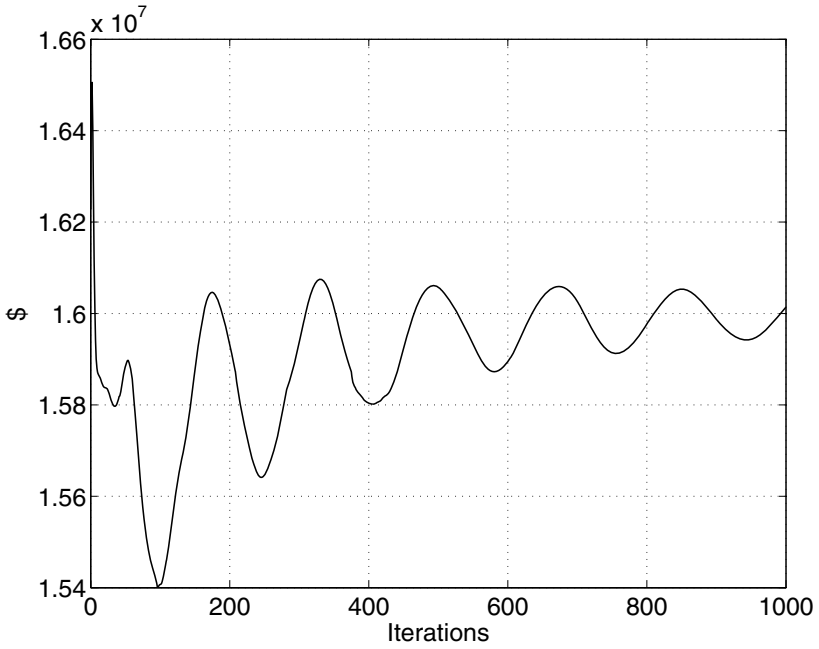

Figure 17: Example 2, the Gauss-Seidel scheme. Profit for the choice of $c = 10^9$

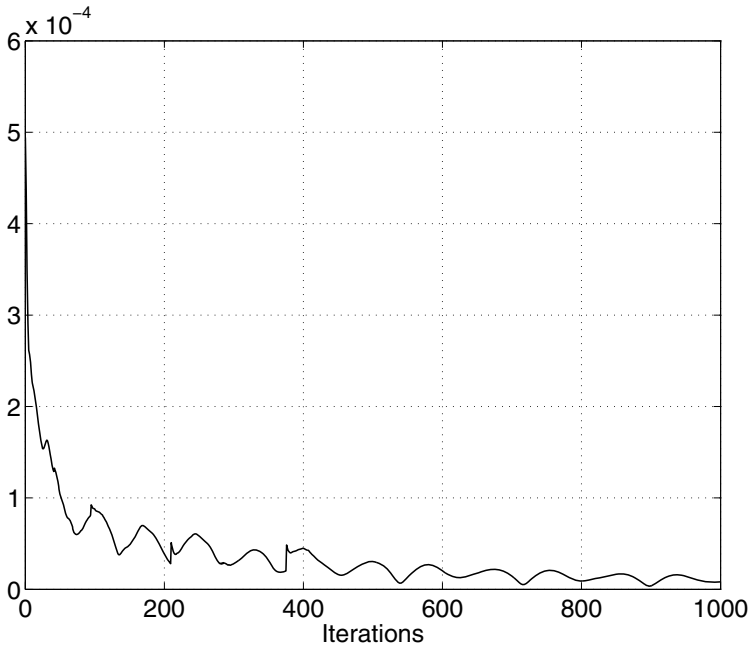

Figure 18: Example 2, the Gauss-Seidel scheme. Residual for the choice of $c = 10^9$

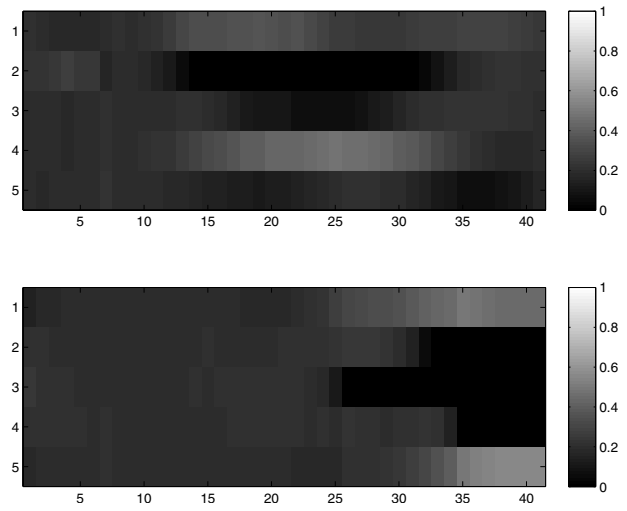

Figure 19: Example 2, the Gauss-Seidel scheme. Optimal rates found for the choice of $c = 10^9$

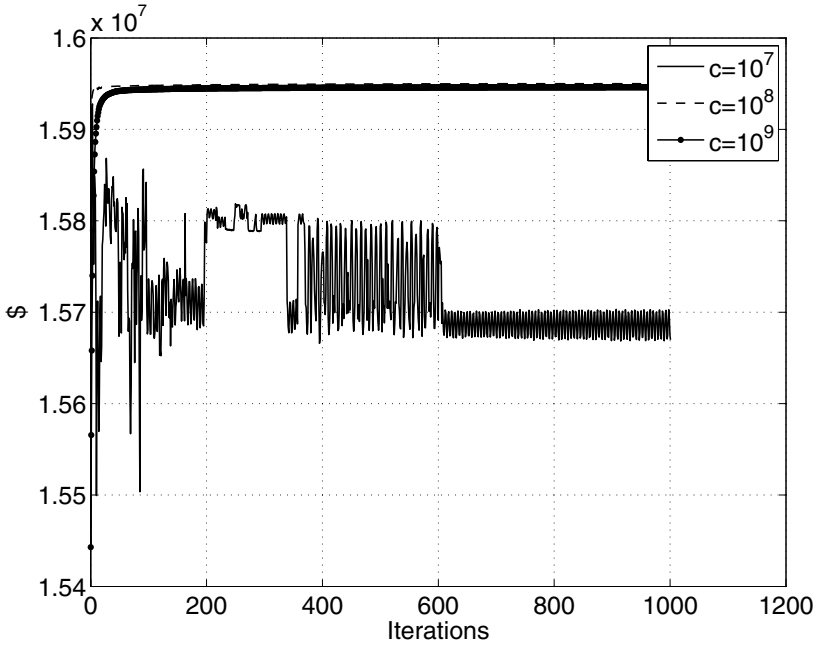

Figure 20: Example 2: Comparisons of the profit obtained with the old marching scheme, with three different values of c.

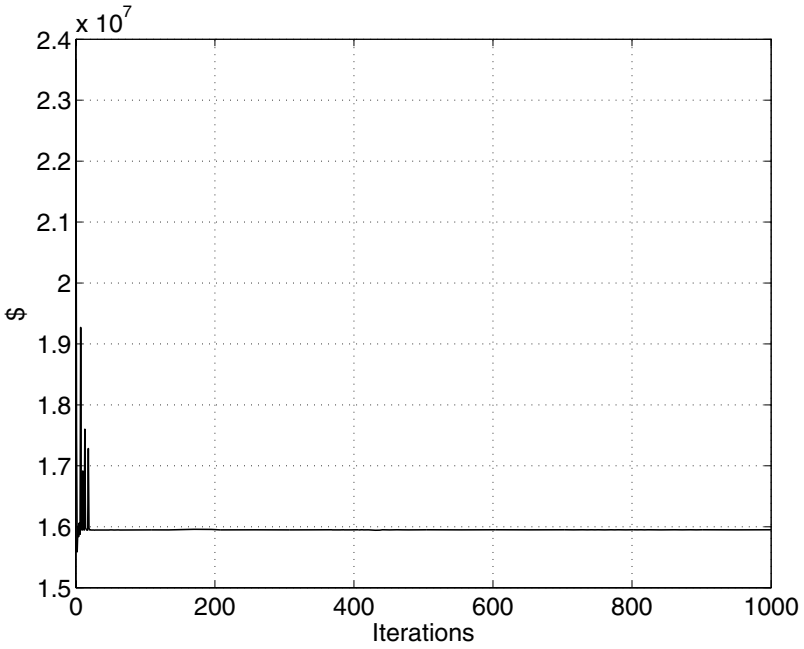

Figure 21: Example 2, the old marching scheme. Profit for the choice of $c = 10^8$

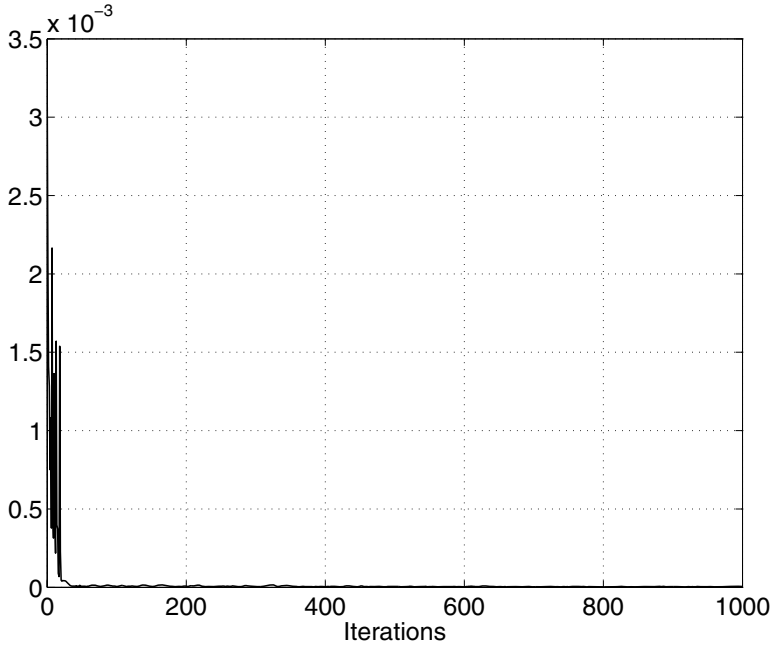

Figure 22: Example 2, the old marching scheme. Residual for the choice of $c = 10^8$

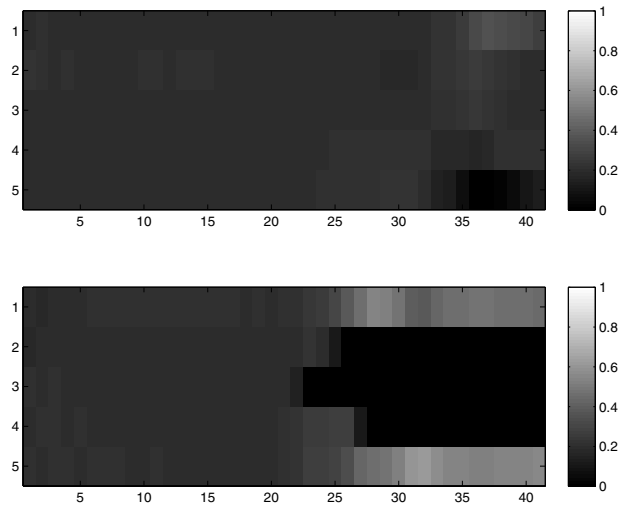

Figure 23: Example 2, the old marching scheme. Optimal rates found for the choice of $c = 10^8$

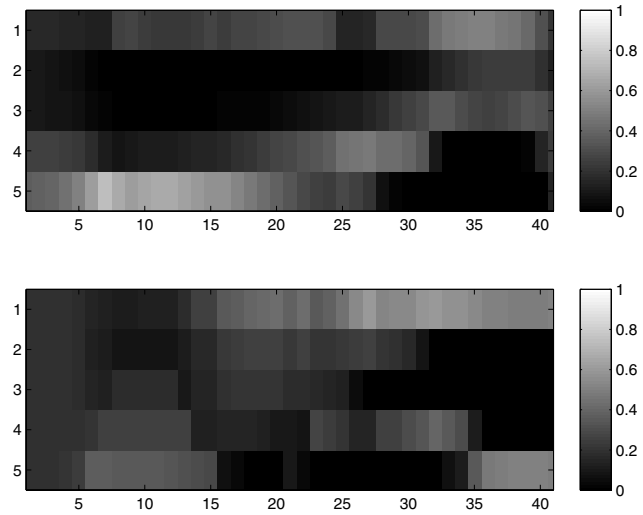

Figure 24: Example 2, the adjoint method. Optimal rates found.

that it is not suited for problems where the objective function is not a least-squares functional. Furthermore, our numerical examples indicate the same problematic behaviour. In our first experiment the old marching scheme either show a very oscillatory behaviour, or it does not converge. For the second experiment, the old marching scheme converges, but it finds a maximum that is much lower than the maximum found with the adjoint method. This illustrates that the method, is not capable of finding the maximum.

Both the new marching scheme and the Gauss-Seidel scheme has shown to be convergent for our examples. In the second example, however, we see from figure 12 and 16 that the new marching scheme converges faster than the Gauss-Seidel scheme. Additionally $\boldsymbol{J}$ for the Gauss-Seidel scheme seems to be more oscillatory than for the new marching scheme.

For problems, where the objective function is not of type least-squares, it seems to better to use the new marching scheme rather than the old marching scheme.

6 Acknowledgements

This work is financed by the Norwegian Research Council, Petromaks programme, project No. 163383/S30, and we are grateful for the support. Furthermore, we want to thank Raymond Martinsen for providing the code of the forward simulator.

References

[1] K. Aziz and A. Settari. *Petroleum Reservoir Simulation*. Kluwer Academic Publishers, 1979.

[2] Dimitri P. Bertsekas. *Constrained Optimisation and Lagrange Multiplier Methods*. Athena Scientific, Belmont, Massachusetts, 1996.

[3] D. R. Brouwer and J.-D. Jansen. Dynamic optimization of waterflooding with smart wells using optimal control theory. *SPE Journal*, 9(4):391–402, 2004.

[4] Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific and Statistical Computing*, 16(5):1190–1208, 1995.

[5] Richard H. Byrd, Jorge Nocedal, and Robert B. Schnabel. Representations of quasi-newton matrices and their use in limited memory methods. *Mathematical Programming 63*, 4:129–156, 1994.

[6] T. F. Chan and X.-C. Tai. Identification of discontinuous coefficients from elliptic problems using total variation regularization. Technical Report CAM 97-35, Dept. of Math., Univeristy of California at Los Angeles, 1997.

[7] Zhiming Chen and Jun Zou. An augmented Lagrangian method for identifying discontinuous parameters in elliptic systems. *SIAM J. Control Optim.*, 37(3):892–910 (electronic), 1999.

[8] Roland Glowinski. *Numerical Methods for Nonlinear Variational Problems*. Springer Verlag, 1984.

[9] Roland Glowinski and Patrick Le Tallec. *Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics*. Society for Industrial and Applied Mathematics, 1989.

[10] Ben-yu Guo and Jun Zou. An augmented Lagrangian method for parameter identifications in parabolic systems. *J. Math. Anal. Appl.*, 263(1):49–68, 2001.

[11] K. Ito, M. Kroller, and K. Kunisch. A numerical study of an augmented Lagrangian method for the estimation of parameters in elliptic systems. *SIAM J. Sci. Statist. Comput.*, 12(4):884–910, 1991.

[12] Kazufumi Ito and Karl Kunisch. The augmented lagrangian method for parameter estimation in elliptic systems. *SIAM Journal of Control and Optimization*, 28(1):113–136, 1990.

[13] Kazufumi Ito and Karl Kunisch. Augmented Lagrangian-SQP methods for nonlinear optimal control problems of tracking type. *SIAM J. Control Optim.*, 34(3):874–891, 1996.

[14] Yee Lo Keung and Jun Zou. Numerical identifications of parameters in parabolic systems. *Inverse Problems*, 14(1):83–100, 1998.

[15] Yee Lo Keung and Jun Zou. An efficient linear solver for nonlinear parameter identification problems. *SIAM J. Sci. Comput.*, 22(5):1511–1526 (electronic), 2000.

[16] K. Kunisch and X.-C. Tai. Sequential and parallel spitting methods for bilinear control problems in hilbert spaces. *SIAM Journal of Numerical Analysis*, 34(1):91–118, 1997.

[17] M. Lien, R. Brouwer, J. D. Jansen, and T. Mannseth. Multiscale regularization of flooding optimization for smart field management. *Paper 99728-MS, in proceedings of the Intelligent Energy Conference and Exhibition, 11-13 April, Amsterdam, The Netherlands*, 2006.

[18] T. K. Nilssen, K. H. Karlsen, T. Mannseth, and X.-C. Tai. Identification of diffusion parameters in a nonlinear convection-diffusion equation using the augmented lagrangian method. *J*, 2003.

[19] Trygve K. Nilssen, Trond Mannseth, and Xue-Cheng Tai. Permeability estimation with the augmented lagrangian method for a nonlinear diffusion equation. *Comput. Geosci.*, 7:27–47, 2003.

[20] Trygve K. Nilssen and Xue-Cheng Tai. Parameter estimation with the augmented lagrangian method for a parabolic equation. *JOTA*, 2005.

[21] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Verlag, 1999.

[22] Pallav Sarma, Khalid Aziz, and Louis J. Durlofsky. Implementation of adjoint solution for optimal control of smart wells. *Paper 92864-MS, in proceedings of the SPE Reservoir Simulation Symposium, 31 January-2 Feburary, The Woodlands, Texas*, 2005.

[23] Iskander S. Zakirov, Sigurd Ivar Aanonsen, Ernest S. Zakirov, and Boris M Palatnik. Optimizing reservoir performance by allocation of well rates. *5th European conference on the Mathematics of Oil Recovery, Leoben, Austrian*, 1996.

A Production Optimisation Problem

We consider a rectangular, heterogeneous, two-dimensional, two-phase (oil and water) reservoir, with no-flow boundaries which is horizontal such that gravitational effects can be ignored. There is located a smart well injector along the left edge of the reservoir, and a smart well producer along the right edge of the reservoir, as shown in figure 25. The two smart wells have several valve openings, indicated by black dots, such that the injection and the production can by controlled individually at each of the valve openings. Initially the reservoir is completely oil saturated, and at the start of operation water is injected in the wells on the left hand side of the reservoir, whilst we are producing from the wells on the right hand side of the reservoir. In the beginning only oil is produced, but after a certain time we will commence to produce both oil and water, and finally only water is produced. There is a profit associated with production of oil. However when producing oil and water simultaneously we must remove the water from the oil, resulting in a cost associated with production of water. The total production rate is fixed, and we assume that we inject at the same rate as we produce. We have the possibility to control the profit by adjusting the percentage of total injection/production in each of the well segments. So our goal is to maximise the NPV from an oil reservoir by adjusting the individual injection and production rates at distinct times. The reservoir flow equations are,

$$-\phi\frac{\partial S_o}{\partial t} - \nabla\cdot\Big(\kappa(x)\lambda_o(S_o)\nabla p_o\Big) = q_o(x), \tag{48}$$

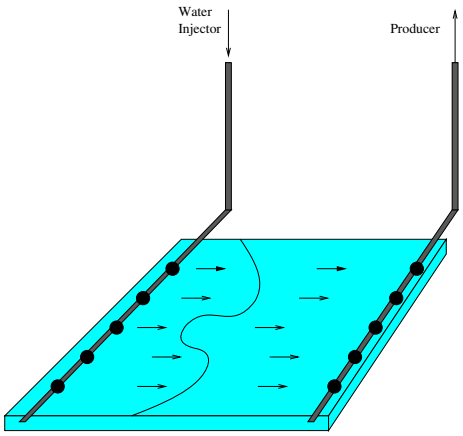

Figure 25: Reservoir.

and

$$-\phi\frac{\partial S_w}{\partial t} - \nabla\cdot\Big(\kappa(x)\lambda_w(S_w)\nabla p_w\Big) = q_w(x), \tag{49}$$

where x is position, t is time, ϕ is porosity, κ is absolute permeability, $\lambda_\alpha = \kappa_{r\alpha}/\mu_\alpha$ is the phase mobility, where $\kappa_{r\alpha}$ is relative permeability and μ_α is viscosity, p_α is pressure, S_α is saturation and the well term, q_α, is flow rate per unit volume, where the subscripts α denotes the fluid phase, o or w. The reservoir is assumed to be fully saturated so that

$$S_w + S_o = 1. \tag{50}$$

From now on, and in the rest of this paper, we will denote the water saturation as S, and the oil saturation as $1-S$. Furthermore we assume that the capillary pressure, $p_c = p_o - p_w$, is zero so that

$$p_w = p_o, \tag{51}$$

and in the rest of this paper, the pressure is denoted by p. In addition we have no-flow boundaries,

$$\frac{\partial p}{\partial \vec{n}} = 0, \tag{52}$$

where $\vec{n}$ is the unit normal vector to the boundary. The relative permeabilities are defined by the Corey models

$$\kappa_{ro} = \kappa^\star_{ro}\Big(\frac{1-S-S_{or}}{1-S_{or}-S_{wr}}\Big)^{e_o} \tag{53}$$

and

$$\kappa_{rw} = \kappa^\star_{rw}\Big(\frac{S-S_{wr}}{1-S_{wr}-S_{or}}\Big)^{e_w}, \tag{54}$$

where e_o and e_w are the Corey exponents, $\kappa^\star_{ro}$ and $\kappa^\star_{rw}$ are the endpoint permeabilities and S_{or} and S_{wr} are the residual saturations, for oil and water respectively.

The equations (48) and (49) are discretised using a standard cell centred grid, with the scheme given by Aziz and Sattari [1], with upstream weighting and using backward Euler to approximate the time derivative. The details can be found in appendix A. From the discretisation we get a discrete time model of the two-phase conservation equations

$$e^n(v^n, u^n, u^{n-1}) = 0, \quad \text{for} \quad n = 1, \dots N, \tag{55}$$

where $e^n = \{e_o^n, e_w^n\}^T$ is the residual column vector corresponding to the discretised equations of (48) and (49) , the superscript n denotes the discrete time step, N is the total number of time steps, $u^n = \{p^n, S^n\}$ is a column vector consisting of the pressures and the water saturations in all the grid cells at time step n, and v^n is the column vector consisting of the control variables, at time step n, which elements are related to the water injection and liquid production rates in the different segments of the wells.

Instead of using a well model, we will directly control water injection and liquid production rates per well segment, such that the wells are treated as source terms. We assume that the total water injection rate equals the total liquid production rate, and we denote the total injection/production rate as V.

Letting v_i^n denote the percent of total injection or production in well segment i at time step n, we have the relations

$$\sum_{i=1}^{N_{inj}} v_i^n = 1, \quad \text{for} \quad n = 1, \dots, N, \tag{56}$$

and

$$\sum_{i=1+N_{inj}}^{N_{prod}+N_{inj}} v_i^n = 1, \quad \text{for} \quad n = 1, \dots, N, \tag{57}$$

where N_{inj} is the number of well segments in the injector and N_{prod} is the number of well segments in the producer. Moreover, the control variables must also satisfy the following inequality constraints, for $n = 1, \dots, N$,

$$0 \leq v_i^n \leq 1, \quad \text{for} \quad i = 1, \dots, N_{inj} + N_{prod}. \tag{58}$$

Since only water is injected, the liquid rate equals the water rate for an injection segment. Thus we have that, for $n = 1, \dots, N$,

$$q_{wi}^n = v_i^n V, \quad \text{for} \quad i = 1, \dots, N_{inj}, \tag{59}$$

where q_{wi}^n is the water rate at injection segment i at time step n. At the production segments, however, the liquid rate equals the sum of the water and oil rates so that, for $n = 1, \dots, N$,

$$q_{wi}^n + q_{oi}^n = -v_i^n V, \quad \text{for} \quad i = 1 + N_{inj}, \dots, N_{prod} + N_{inj}, \tag{60}$$

where q_{wi}^n and q_{oi}^n are the water and oil rates, respectively, at production segment i at time step n. The different phase production rates can be expressed as functions of the liquid rate and the fractional flow at the well segment so that

$$q_{wi}^n = -\frac{\lambda_{wi}^n}{\lambda_{wi}^n + \lambda_{oi}^n} \, v_i^n V \quad \text{for} \quad i = 1 + N_{inj}, \ldots, N_{prod} + N_{inj}, \tag{61}$$

and

$$q_{oi}^n = -\frac{\lambda_{oi}^n}{\lambda_{oi}^n + \lambda_{wi}^n} \, v_i^n V, \quad \text{for} \quad i = 1 + N_{inj}, \ldots, N_{prod} + N_{inj}, \tag{62}$$

where λ_{wi}^n and λ_{oi}^n are the water and oil mobilities, respectively, in the grid cell containing well segment i and at time step n.

A.1 Profit Function

Our aim is to maximise net present value, by controlling the individual well segment rates during the entire production period. The net present value (NPV), J, is given as

$$J = \sum_{n=1}^{N} J^n, \tag{63}$$

with

$$J^n = \Delta x \Delta y h \left[\sum_{i=1+N_{inj}}^{N_{prod}+N_{inj}} \frac{-I_w \cdot q_{wi}^n - I_o \cdot q_{oi}^n}{(1 + b/100)^{t^n}} \right] \Delta t^n, \tag{64}$$

where the constants I_o and I_w are, respectively, the revenue of oil produced and the cost of water produced per volume expressed in $\$/m^3$, b is the annual interest rate expressed in %, Δx and Δy are the dimensions of the grid cells in, respectively, horizontal and vertical direction, Δt^n is the size of the n'th time step, and $t^n = \sum_{i=1}^{n} \Delta t^i$ is the time expressed in years at time step n. Since the production rates of water and oil, q_{wi}^n and q_{oi}^n, are less than or equal to zero, I_o is a positive constant and I_w is a negative constant. The objective is to maximise the function (63) by adjusting the percentage of total injection and production in each of the individual segments of, respectively, the injection well and the production well.

B The Forward model

In order to discretise the flow equations (48) and (49), we use a cell centred finite difference method, where the backward Euler method is used to discretise the time derivative. We divide our reservoir into a discrete number of grid cells as shown in figure 26, where we assume that all variables are constant within each cell. The length of the vertical edge of cell $\{i, j\}$ is Δy_j and its horizontal length is Δx_i. Furthermore we divide the time line into a discrete set of N time intervals, where the length of time interval number n is denoted by Δt^n. We want that our equations

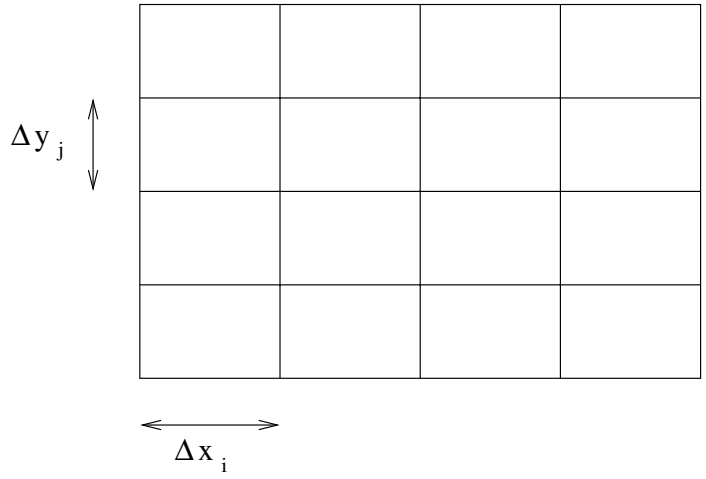

Figure 26: Grid

should be on dimensionless form, and therefore we multiply by the discrete time step Δt^n and we find that the discrete approximation of the flow equations for cell $\{i, j\}$ and phase $\alpha \in \{w, o\}$ is given by

$$\phi_{i,j}(S^n_{i,j,\alpha} - S^{n-1}_{i,j,\alpha}) - \Delta t^n \Phi^n_{i,j,\alpha} = \Delta t^n q^n_{i,j,\alpha}. \tag{65}$$

where the state variables $u^n_{i,j} = \{p^n_{i,j}, S^n_{i,j,w}\}$, and the operator $\Phi^n_{i,j,\alpha}$ is a discrete approximation to the second term in equations (48) and (49). The discrete operator $\Phi^n_{i,j,\alpha}$ is discretised according to a formula from [1]. When omitting the subscript α, it gives that

$$\Phi^n_{i,j} = \left(\lambda_{i+\frac{1}{2},j} T_{i+\frac{1}{2},j}(p^n_{i+1,j} - p^n_{i,j}) - \lambda_{i-\frac{1}{2},j} T_{i-\frac{1}{2},j}(p^n_{i,j} - p^n_{i-1,j})\right)/\Delta x_i \tag{66}$$

$$+ \left(\lambda_{i,j+\frac{1}{2}} T_{i,j+\frac{1}{2}}(p^n_{i,j+1} - p^n_{i,j}) - \lambda_{i,j-\frac{1}{2}} T_{i,j-\frac{1}{2}}(p^n_{i,j} - p^n_{i,j-1})\right)/\Delta y_j, \tag{67}$$

where the transmisibilities T are given by

$$T_{i+\frac{1}{2},j} = \frac{1}{\frac{1}{2}\left(\frac{\Delta x_i}{\kappa_{i,j}} + \frac{\Delta x_{i+1}}{\kappa_{i+1,j}}\right)}, \tag{68}$$

$$T_{i-\frac{1}{2},j} = \frac{1}{\frac{1}{2}\left(\frac{\Delta x_i}{\kappa_{i,j}} + \frac{\Delta x_{i-1}}{\kappa_{i-1,j}}\right)}, \tag{69}$$

$$T_{i,j+\frac{1}{2}} = \frac{1}{\frac{1}{2}\left(\frac{\Delta y_j}{\kappa_{i,j}} + \frac{\Delta y_{j+1}}{\kappa_{i,j+1}}\right)}, \tag{70}$$

$$T_{i,j-\frac{1}{2}} = \frac{1}{\frac{1}{2}\left(\frac{\Delta y_j}{\kappa_{i,j}} + \frac{\Delta y_{j-1}}{\kappa_{i,j-1}}\right)}. \tag{71}$$

For the mobilities λ, we use upstream weighting. That is

$$\lambda_{i+\frac{1}{2},j} = \begin{cases} \lambda_{i+1,j}, & p_{i+1,j} > p_{i,j} \\ \lambda_{i,j}, & p_{i+1,j} \leq p_{i,j} \end{cases} \tag{72}$$

$$\lambda_{i-\frac{1}{2},j} = \begin{cases} \lambda_{i-1,j}, & p_{i-1,j} > p_{i,j} \\ \lambda_{i,j}, & p_{i-1,j} \leq p_{i,j} \end{cases} \tag{73}$$

$$\lambda_{i,j+\frac{1}{2}} = \begin{cases} \lambda_{i,j+1}, & p_{i,j+1} > p_{i,j} \\ \lambda_{i,j}, & p_{i,j+1} \leq p_{i,j} \end{cases} \tag{74}$$

$$\lambda_{i,j-\frac{1}{2}} = \begin{cases} \lambda_{i,j-1}, & p_{i,j-1} > p_{i,j} \\ \lambda_{i,j}, & p_{i,j-1} \leq p_{i,j} \end{cases} \tag{75}$$

Furthermore, we assume that the liquids are incompressible, and let the total injection/production rate, V, be constant, not varying with time. Since we only inject water, the injection rate at well segment i is

$$q_{wi}^n = v_i^n V, \quad \text{for} \quad i = 1, \ldots, N_{inj}, \tag{76}$$

where q_{wi}^n is the water rate at injection segment i at time step n. The different phase production rates can be expressed as functions of the liquid rate and the fractional flow at the well segment so that

$$q_{wi}^n = -\frac{\lambda_{wi}^n}{\lambda_{wi}^n + \lambda_{oi}^n} v_i^n V \quad \text{for} \quad i = 1 + N_{inj}, \ldots, N_{prod} + N_{inj}, \tag{77}$$

and

$$q_{oi}^n = -\frac{\lambda_{oi}^n}{\lambda_{oi}^n + \lambda_{wi}^n} v_i^n V, \quad \text{for} \quad i = 1 + N_{inj}, \ldots, N_{prod} + N_{inj}, \tag{78}$$

where λ_{wi}^n and λ_{oi}^n are the water and oil mobilities, respectively, in the grid cell containing well segment i and at time step n. The discrete approximations to the reservoir flow equations for time step n (65) can be written on vectorial form as

$$e^n(v^n, u^n, u^{n-1}) = \begin{Bmatrix} e_o^n \\ e_w^n \end{Bmatrix} = 0 \tag{79}$$

where the e^n is the equation residual, and the subscripts w and o denotes the equation residual for the flow equations for oil and water, respectively. Now we have that

$$e_\alpha^n(v^n, u^n, u^{n-1}) = \phi(S_\alpha^n - S_\alpha^{n-1}) - \Delta t^n \Phi_\alpha^n - \Delta t^n q_\alpha^n, \tag{80}$$

where this vector equation is organised so that the flow equation for grid cell $\{1, 1\}$ is its first element, grid cell $\{n_x, 1\}$ is element n_x in the vector and grid cell $\{n_x, n_y\}$ is the last element.

C The Derivatives

In order to solve the forward problem and to perform optimisation we need to calculate derivatives.

First we will investigate the derivatives of the equation residual e with respect to the state variables $u = \{p, S\}$.

C.1 The Derivatives with respect to Pressure and Saturation

Even though we have the equation residual written on vectorial form, we will use grid notation in this subsection, since this makes it easier to describe the derivatives. For the derivative with respect to pressure, we have that

$$\frac{\partial e^n}{\partial p_{i,j}} = \frac{\partial e^n_{i,j}}{\partial p_{i,j}} + \frac{\partial e^n_{i+1,j}}{\partial p_{i,j}} + \frac{\partial e^n_{i-1,j}}{\partial p_{i,j}} + \frac{\partial e^n_{i,j+1}}{\partial p_{i,j}} + \frac{\partial e^n_{i,j-1}}{\partial p_{i,j}}, \tag{81}$$

where

$$\frac{\partial e^n_{i,j}}{\partial p_{i,j}} = -\Delta t^n \left((\lambda_{i+\frac{1}{2},j} T_{i+\frac{1}{2},j} + \lambda_{i-\frac{1}{2},j} T_{i-\frac{1}{2},j})/\Delta x_i + (\lambda_{i,j+\frac{1}{2}} T_{i,j+\frac{1}{2}} + \lambda_{i,j-\frac{1}{2}} T_{i,j-\frac{1}{2}})/\Delta y_j \right), \tag{82}$$

$$\frac{\partial e^n_{i+1,j}}{\partial p_{i,j}} = \Delta t^n \lambda_{i+\frac{1}{2},j} T_{i+\frac{1}{2},j}/\Delta x_{i+1}, \tag{83}$$

$$\frac{\partial e^n_{i-1,j}}{\partial p_{i,j}} = \Delta t^n \lambda_{i-\frac{1}{2},j} T_{i-\frac{1}{2},j}/\Delta x_{i-1}, \tag{84}$$

$$\frac{\partial e^n_{i,j+1}}{\partial p_{i,j}} = \Delta t^n \lambda_{i,j+\frac{1}{2}} T_{i,j+\frac{1}{2}}/\Delta y_{j+1}, \tag{85}$$

$$\frac{\partial e^n_{i,j-1}}{\partial p_{i,j}} = \Delta t^n \lambda_{i,j-\frac{1}{2}} T_{i,j-\frac{1}{2}}/\Delta y_{j-1}. \tag{86}$$

Furthermore, the derivative with respect to saturation is given as

$$\frac{\partial e^n}{\partial S^n_{i,j}} = \frac{\partial e^n_{i,j}}{\partial S^n_{i,j}} + \frac{\partial e^n_{i+1,j}}{\partial S^n_{i,j}} + \frac{\partial e^n_{i-1,j}}{\partial S^n_{i,j}} + \frac{\partial e^n_{i,j+1}}{\partial S^n_{i,j}} + \frac{\partial e^n_{i,j-1}}{\partial S^n_{i,j}}, \tag{87}$$

where

$$\frac{\partial e^n_{i,j}}{\partial S^n_{i,j}} = \phi_{i,j} - \Delta t^n \frac{\partial \Phi_{i,j}}{\partial S^n_{i,j}} - \Delta t^n \frac{\partial q_{i,j}}{\partial S^n_{i,j}} \tag{88}$$

$$\frac{\partial e^n_{i+1,j}}{\partial S^n_{i,j}} = -\Delta t^n \frac{\partial \Phi_{i+1,j}}{\partial S^n_{i,j}} - \Delta t^n \frac{\partial q_{i+1,j}}{\partial S^n_{i,j}} \tag{89}$$

$$\frac{\partial e^n_{i-1,j}}{\partial S^n_{i,j}} = -\Delta t^n \frac{\partial \Phi_{i-1,j}}{\partial S^n_{i,j}} - \Delta t^n \frac{\partial q_{i-1,j}}{\partial S^n_{i,j}} \tag{90}$$

$$\frac{\partial e^n_{i,j+1}}{\partial S^n_{i,j}} = -\Delta t^n \frac{\partial \Phi_{i,j+1}}{\partial S^n_{i,j}} - \Delta t^n \frac{\partial q_{i,j+1}}{\partial S^n_{i,j}} \tag{91}$$

$$\frac{\partial e^n_{i,j-1}}{\partial S^n_{i,j}} = -\Delta t^n \frac{\partial \Phi_{i,j-1}}{\partial S^n_{i,j}} - \Delta t^n \frac{\partial q_{i,j-1}}{\partial S^n_{i,j}}. \tag{92}$$

In the expression for $\Phi_{i,j}$ (66), it is only the mobilities are functions of the saturation, so to find the derivative of $\Phi_{i,j}$ with respect to the saturation, one simply uses the chain rule, and the derivatives of the mobilities with respect to saturation is when omitting the superscript n,

$$\frac{\partial \lambda_{i\alpha}}{\partial S_i} = \frac{1}{\mu_\alpha} \frac{\partial \kappa_{r\alpha}}{\partial S_i}, \tag{93}$$

where

$$\frac{\partial \kappa_{ro}}{\partial S_i} = -e_o \kappa^\star_{ro} \frac{(1 - S_i - S_{or})^{e_o - 1}}{(1 - S_{or} - S_{wr})^{e_o}} \tag{94}$$

and

$$\frac{\partial \kappa_{rw}}{\partial S_i} = e_w \kappa^\star_{rw} \frac{(S_i - S_{wr})^{e_w - 1}}{(1 - S_{or} - S_{wr})^{e_w}}. \tag{95}$$

For the source terms we have that

$$\frac{\partial q_{iw}}{\partial S_i} = \left[\frac{\partial \lambda_{io}}{\partial S_i} \lambda_{iw} - \frac{\partial \lambda_{iw}}{\partial S_i} \lambda_{io} \right] / (\lambda_{iw} + \lambda_{io})^2, \tag{96}$$

and

$$\frac{\partial q_{io}}{\partial S_i} = \left[\frac{\partial \lambda_{iw}}{\partial S_i} \lambda_{io} - \frac{\partial \lambda_{io}}{\partial S_i} \lambda_{iw} \right] / (\lambda_{io} + \lambda_{iw})^2. \tag{97}$$

C.2 Derivatives with respect to the state variables from the previous time step

Regarding the derivatives of the equation residual, with respect to the state variables from the previous time step, we have that

$$\frac{\partial e^n}{\partial u^{n-1}} = \begin{pmatrix} \frac{\partial e^n_1}{\partial p^{n-1}_1} & \cdots & \frac{\partial e^n_1}{\partial p^{n-1}_m} & \frac{\partial e^n_1}{\partial s^{n-1}_1} & \cdots & \frac{\partial e^n_1}{\partial s^{n-1}_m} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e^n_m}{\partial p^{n-1}_1} & \cdots & \frac{\partial e^n_m}{\partial p^{n-1}_m} & \frac{\partial e^n_m}{\partial s^{n-1}_1} & \cdots & \frac{\partial e^n_m}{\partial s^{n-1}_m} \\ \frac{\partial e^n_{m+1}}{\partial p^{n-1}_1} & \cdots & \frac{\partial e^n_{m+1}}{\partial p^{n-1}_m} & \frac{\partial e^n_{m+1}}{\partial s^{n-1}_1} & \cdots & \frac{\partial e^n_{m+1}}{\partial s^{n-1}_m} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e^n_{2m}}{\partial p^{n-1}_1} & \cdots & \frac{\partial e^n_{2m}}{\partial p^{n-1}_m} & \frac{\partial e^n_{2m}}{\partial s^{n-1}_1} & \cdots & \frac{\partial e^n_{2m}}{\partial s^{n-1}_m} \end{pmatrix}, \tag{98}$$

where

$$\frac{\partial e^n_i}{\partial s^{n-1}_j} = \begin{cases} \phi_j & \text{if } i = j \\ -\phi_j & \text{if } i = j + m \\ 0 & \text{else} \end{cases} \tag{99}$$

This leads to

$$\frac{\partial e^n}{\partial u^{n-1}} = \begin{pmatrix} 0 & \dots & \dots & 0 & \phi_1 & 0 & \dots & 0 \\ \vdots & \ddots & & \vdots & 0 & \phi_2 & & \vdots \\ \vdots & & \ddots & \vdots & \vdots & & \ddots & 0 \\ 0 & \dots & \dots & 0 & 0 & \dots & 0 & \phi_m \\ 0 & \dots & \dots & 0 & -\phi_1 & 0 & \dots & 0 \\ \vdots & \ddots & & \vdots & 0 & -\phi_2 & & \vdots \\ \vdots & & \ddots & \vdots & \vdots & & \ddots & 0 \\ 0 & \dots & \dots & 0 & 0 & \dots & 0 & -\phi_m \end{pmatrix}, \tag{100}$$

since the equation is independent of the pressures from the previous time step.

C.3 The derivatives with respect to the controls

With regards to the controls, they are in the production optimisation problem the percentages of total injection/production in each of the different wells, denoted as v_i for well i. The only part of the residual equations that depend on v is the well terms, and it is given by

$$\frac{\partial e_{oi}^n}{\partial v_j^n} = \frac{\partial q_{oi}^n}{\partial v_j^n} = \begin{cases} -\frac{\lambda_{oi}^n}{\lambda_{oi}^n + \lambda_{wi}^n} V & \text{if} \quad i = j \\ 0 & \text{else} \end{cases} \tag{101}$$

and

$$\frac{\partial e_{wi}^n}{\partial v_j^n} = \frac{\partial q_{wi}^n}{\partial v_j^n} = \begin{cases} -\frac{\lambda_{wi}^n}{\lambda_{oi}^n + \lambda_{wi}^n} V & \text{if} \quad i = j \\ 0 & \text{else} \end{cases} \tag{102}$$

Paper C

Efficient History Matching and Production Optimisation with the Augmented Lagrangian Method.

SPE 105833

Efficient History Matching and Production Optimization with the Augmented Lagrangian Method

D. C. Doublet, S.I. Aanonsen, SPE, and X.-C. Tai, Centre for Integrated Petroleum Research, U. of Bergen

This paper was prepared for presentation at the 2007 SPE Reservoir Simulation Symposium held in Houston, Texas, U.S.A., 26–28 February 2007.

Abstract

A new method for reservoir history matching and dynamic optimization of water flooding with smart wells is presented. For the dynamic optimization of water flooding case, we solve a constrained optimization problem where the net present value is maximized and the reservoir flow equations are considered as constraints. Similarly for the history matching, we recover the permeability field from measurements of pressures and saturations. In both cases the problem is formulated as finding a saddle point of the associated augmented Lagrangian functional. We compare the method with a more traditional optimal control method, based on solving the adjoint system of equations. In the examples tested the new method obtains approximately the same results using the same or less computational effort compared to the adjoint method. An advantage of the new method is that we do not solve the flow equations exactly at each iteration. As the optimization proceeds, the flow equations will be fulfilled at convergence. Thus, each iteration of the new minimization algorithm is cheaper than for the adjoint method.

Paper D

Efficient Optimisation of Production from Smart Wells Based on the Augmented Lagrangian Method

Efficient optimization of production from smart wells based on the augmented Lagrangian method.

D. C. Doublet, R. Martinsen, S. I. Aanonsen, X.-C. Tai

Abstract

A new method for dynamic optimisation of water flooding with smart wells is developed. The algorithm finds optimal injection and production well or well segment rates. In the new method, we solve a constrained optimisation problem where the net present value is maximised and the reservoir flow equations are considered as constraints. The problem is formulated as finding the saddle point of the associated augmented Lagrangian functional, and solved efficiently. The method is compared with a more traditional optimal-control method, based on solving the adjoint system of equations. In the examples tested the new method obtains the same maximum profit as the adjoint method using approximately the same number of iterations. An advantage of the new method is that we do not solve the flow equations exactly at each iteration. As the optimisation proceeds, the flow equations will be fulfilled at convergence. Thus, each iteration of the minimisation algorithm is much cheaper than for the adjoint method. The method is tested on a small 2D model, but the results should be valid also for larger, 3D models.

Introduction

In this paper we propose a new method for maximising the net present value (NPV) of an oil reservoir, by reducing water production and increasing oil recovery at the same time as we are delaying water breakthrough. Optimal control theory methods has earlier been used to solve this problem in e.g. Brouwer and Jansen (2004), Lien et al. (2006), Sarma et al. (2005) and Zakirov et al. (1996). In this article we formulate the optimisation problem as an augmented Lagrangian saddle point problem and present a new method for solving it. The effectiveness of the method is demonstrated in numerical examples, and we also present comparisons with the adjoint method. We consider a horizontal, two-dimensional, two-phase (oil and water) reservoir, with no-flow boundaries and no capillary pressure. Two horizontal smart wells, one injector and one producer, are located at opposite sides of the reservoir. This is shown in figure 1. The reservoir flow equations are,

$$-\phi\frac{\partial S_o}{\partial t}-\nabla\cdot\Big(\kappa(x)\lambda_o(S_o)\nabla p\Big)=q_o(x),\quad -\phi\frac{\partial S_w}{\partial t}-\nabla\cdot\Big(\kappa(x)\lambda_w(S_w)\nabla p\Big)=q_w(x), \tag{1}$$

where x is position, t is time, ϕ is porosity, κ is absolute permeability, $\lambda_{r\alpha}=\kappa_{r\alpha}/\mu_\alpha$ is the phase mobility, where $\kappa_{r\alpha}$ is relative permeability and μ_α is viscosity, p is pressure, S_α is saturation and the well term, q_α, is flow rate per unit volume, where the subscripts α denotes the fluid phase, o or w. The equations (1) are discretised using a standard cell centred grid with upstream weighting and using backward Euler to approximate the time derivative. From the discretisation we get a discrete time model of the two-phase conservation equations

$$e^n(u^n,u^{n-1},v^n)=0,\quad \text{for}\quad n=1,\dots,N, \tag{2}$$

where e is a nonlinear vector function, the superscript n denotes the discrete time step, N is the total number of time steps, $u^n=\{p^n,S^n\}$ is a vector consisting of pressures and water saturations in all the grid cells at time step n, and v^n is the vector consisting of the control variables at time step n, whose elements are related to the water injection and liquid production rates in the different segments of the wells. Instead of using a well model, we will directly control water injection and liquid production rates per well segment.

We assume that the total water injection rate equals the total liquid production rate, and we denote the total injection/production rate as V. Letting v_i^n denote the percent of total injection or production in well segment i at time step n, we have the relations

$$\sum_{i=1}^{N_{inj}} v_i^n=1,\qquad \sum_{i=1+N_{inj}}^{N_{prod}+N_{inj}} v_i^n=1,\quad \text{for}\quad n=1,\dots,N, \tag{3}$$

where N_{inj} is the number of well segments in the injector and N_{prod} is the number of well segments in the producer. Moreover, the control variables must also satisfy the following inequality constraints, for $n=1,\dots,N$,

$$0\le v_i^n\le 1,\quad \text{for}\quad i=1,\dots,N_{inj}+N_{prod}. \tag{4}$$

Since only water is injected, the liquid rate equals the water rate for an injection segment. Thus we have that, for $n=1,\dots,N$,

$$q_{wi}^n=v_i^nV,\quad \text{for}\quad i=1,\dots,N_{inj}, \tag{5}$$

where q_{wi}^n is the water rate at injection segment i at time step n. At the production segments, however, the liquid rate equals the sum of the water and oil rates so that, for $n=1,\dots,N$,

$$q_{wi}^n+q_{oi}^n=-v_i^nV,\quad \text{for}\quad i=1+N_{inj},\dots,N_{prod}+N_{inj}, \tag{6}$$

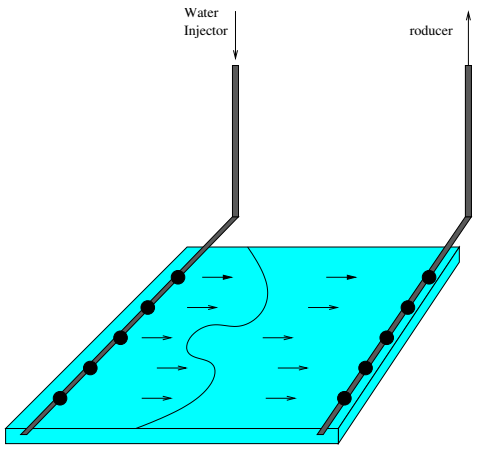

Figure 1: Reservoir.

where q^n_{wi} and q^n_{oi} are the water and oil rates, respectively, at production segment i at time step n. The different phase production rates can be expressed as functions of the liquid rate and the fractional flow at the well segment so that

$$q^n_{wi} = -\frac{\lambda^n_{wi}}{\lambda^n_{wi}+\lambda^n_{oi}}\, v^n_i V \quad \text{and} \quad q^n_{oi} = -\frac{\lambda^n_{oi}}{\lambda^n_{oi}+\lambda^n_{wi}}\, v^n_i V, \quad \text{for} \quad i = 1+N_{inj},\ldots,N_{prod}+N_{inj}, \tag{7}$$

where λ^n_{wi} and λ^n_{oi} are the water and oil mobilities, respectively, in the grid cell containing well segment i and at time step n.

Profit Function

Our aim is to maximise net present value, by controlling the individual well segment rates during the entire production period. The net present value (NPV), J, is given as

$$J = \sum_{n=1}^{N} J^n, \quad \text{with} \quad J^n = \Delta x \Delta y h \left[\sum_{i=1+N_{inj}}^{N_{prod}+N_{inj}} \frac{-I_w \cdot q^n_{wi} - I_o \cdot q^n_{oi}}{(1+b/100)^{t^n}} \right] \Delta t^n, \tag{8}$$

where the constants I_o and I_w are, respectively, the revenue of oil produced and the cost of water produced per volume expressed in $\$/m^3$, b is the annual interest rate expressed in %, Δx and Δy are the dimensions of the grid cells in, respectively, horizontal and vertical direction, Δt^n is the size of the n'th time step, and $t^n = \sum_{i=1}^{n} \Delta t^i$ is the time expressed in years at time step n. Since the production rates of water and oil, q^n_{wi} and q^n_{oi}, are less than or equal to zero, I_o is a positive constant and I_w is a negative constant.

The objective is to maximise the function (8) by adjusting the percentage of total injection and production in each of the individual segments of, respectively, the injection well and the production well.

Saddle Point Problem Formulation

Let us define the control variables at time step n as $v^n = \{v^n_i\}_{i=1}^{N_{inj}+N_{prod}}$, and let $v = \{v^n\}_{n=1}^{N}$ and $u = \{u^n\}_{n=1}^{N}$. We can now formulate our problem as a constrained minimisation problem where we want to solve

$$\min_{v,u} -J(v,u) \qquad \text{subject to} \quad e^n(v^n,u^n,u^{n-1}) = 0, \quad \text{for } n = 1,\ldots,N. \tag{9}$$

Let us now define the Lagrangian by

$$L(v,u,\lambda) = \sum_{n=1}^{N} L^n(v^n,u^n,u^{n-1},\lambda^n), \tag{10}$$

where

$$L^n(v^n,u^n,u^{n-1},\lambda^n) = -J^n(v^n,u^n) + \lambda^{nT} e^n(v^n,u^n,u^{n-1}), \tag{11}$$

and $\lambda = \{\lambda^n\}_{n=1}^N$ are the Lagrangian multipliers. Furthermore, we define the augmented Lagrangian by

$$L_c(v,u,\lambda) = \sum_{n=1}^{N} L_c^n(v^n,u^n,u^{n-1},\lambda^n), \tag{12}$$

where

$$L_c^n(v^n,u^n,u^{n-1},\lambda^n) = L^n(v^n,u^n,u^{n-1},\lambda^n) + \frac{c}{2} e^n(v^n,u^n,u^{n-1})^T e^n(v^n,u^n,u^{n-1}), \tag{13}$$

and $c > 0$ is a penalisation constant. It is known that the solution of (9) is a saddle point of (10), and that the saddle point of (10) is also a saddle point of (12), see Glowinski (1984) for proof. Given that the set of constraint gradients is linearly independent at the solution of (9), the following conditions hold at the solution of (9), see e.g. Nocedal and Wright (1999) for proof.

Karush Kuhn Tucker (KKT) conditions Suppose that $\{v^\star,u^\star\}$ is a solution of (9). Then there exists a set of vectors $\lambda^\star$ such that the following conditions are fulfilled at the point $\{v^\star,u^\star\}$,

$$\begin{aligned} \nabla_{u^n} L(v^\star,u^\star,\lambda^\star) &= 0, \\ \nabla_{v^n} L(v^\star,u^\star,\lambda^\star) &= 0, \\ \nabla_{\lambda^n} L(v^\star,u^\star,\lambda^\star) &= 0, \end{aligned} \tag{14}$$

for $n = 1,\ldots,N$. Since the last of these conditions yields $e^n = 0$, for $n = 1,\ldots,N$, it is easy to see that the KKT conditions are equivalent to

$$\begin{aligned} \nabla_{u^n} L_c(v^\star,u^\star,\lambda^\star) &= 0, \\ \nabla_{v^n} L_c(v^\star,u^\star,\lambda^\star) &= 0, \\ \nabla_{\lambda^n} L_c(v^\star,u^\star,\lambda^\star) &= 0, \end{aligned} \tag{15}$$

for $n = 1,\ldots,N$. In order to find the solution of (9), we can thus solve the KKT system (14) or equivalently (15). We propose a new algorithm to solve (9) by solving the system of equations (15).

Optimisation method

Letting the subscript k be the outer iteration counter, we propose the following algorithm to solve the KKT conditions(15).

New KKT Optimisation Algorithm

1. Choose $v_0 = \{v_0^n\}_{n=1}^N$, $u_0 = \{u_0^n\}_{n=1}^N$ and $c > 0$. For $k = 1,2,\ldots$ do:
2. Find λ_k^N such that

$$-\frac{\partial J^N}{\partial u^N}(v_{k-1}^N,u_{k-1}^N) + \left(\lambda_k^N + c\cdot e^N(v_{k-1}^N,u_{k-1}^N,u_{k-1}^{N-1})\right)^T \frac{\partial e^N}{\partial u^N}(v_{k-1}^N,u_{k-1}^N) = 0.$$

3. For $n = N-1, N-2, \ldots, 1$, find λ_k^n, such that

$$-\frac{\partial J^n}{\partial u^n}(v_{k-1}^n, u_{k-1}^n) + \left(\lambda_k^n + c \cdot e^n(v_{k-1}^n, u_{k-1}^n, u_{k-1}^{n-1})\right)^T \frac{\partial e^n}{\partial u^n}(v_{k-1}^n, u_{k-1}^n) + \left(\lambda_k^{n+1} + c \cdot e^{n+1}(v_{k-1}^{n+1}, u_{k-1}^{n+1}, u_{k-1}^n)\right)^T \frac{\partial e^{n+1}}{\partial u^n} = 0.$$

4. Then, for $n = 1, 2, \ldots, N$,we will find v_k^n such that

$$v_k^n = \underset{v^n}{\operatorname{argmin}} \left[-J^n(v^n, u_{k-1}^n) + \left(\lambda_k^n + \frac{c}{2} e^n(v^n, u_{k-1}^n, u_{k-1}^{n-1})\right)^T e^n(v^n, u_{k-1}^n, u_{k-1}^{n-1}) \right].$$

5. And finally update the state variables u^n by

$$u_k^n = u_{k-1}^n - \left(\frac{\partial e^n}{\partial u^n}(v_k^n, u_{k-1}^n)\right)^{-1} e^n(v_k^n, u_{k-1}^n, u_{k-1}^{n-1}).$$

After performing steps 2 and 3 of the algorithm, the first of the KKT conditions are fulfilled. Furthermore we observe that, when calculating v_k from step 4 in the algorithm above, we find v_k such that

$$\nabla_{v^n} L_c(v_k, u_{k-1}, \lambda_k) = 0,$$

for $n = 1, \ldots, N$. Thus the second of the KKT conditions are fulfilled after performing step 4. The forward problem, given by the N non-linear systems of equations $e^n(v^n, u^n, u^{n-1}) = 0$ for $n = 1, \ldots, N$, corresponds to the third of the KKT conditions (15). These may be solved by Newton's method when the initial guess is close to the solution, and step 5 of the new KKT algorithm corresponds to doing one iteration of Newton's method on the forward problem. In this way, we hope that the third of the KKT conditions is fulfilled at convergence. Since the two other KKT conditions are fulfilled at all times, we have then found the solution.

When finding λ^n, for $n = N, \ldots, 1$, in steps 2 and 3 we need to solve a linear system of equations, and we do this with the GMRES method, as described in for example Golub and Van Loan (1996) and Saad (2000). For the minimisation in step 4, we use the LBFGS method which is a quasi-Newton method. For more information on this method see Byrd et al. (1994) and Byrd et al. (1995). Finally for the update in step 5, we use as in steps 3 and 4 the GMRES method to solve the linear system of equations.

Comparison With The Adjoint Method

To compare the results of the new KKT algorithm, we use an optimal control theory method, the adjoint method, used previously in Brouwer and Jansen (2004), Lien et al. (2006), Sarma et al. (2005) and Zakirov et al. (1996), to solve the production optimisation problem. Letting, as for the new KKT algorithm, the subscript k be the outer iteration counter, the adjoint algorithm is as follows.

The Adjoint Algorithm

1. Choose $v_0 = \{v_0^n\}_{n=1}^N$, u_0^0 and $c > 0$. For $k = 1, 2, \ldots$ do:

2. For $n = 1, 2, \ldots, N$, find u_k^n such that

$$e^n(v_{k-1}^n, u_k^n, u_k^{n-1}) = 0.$$

3. Find λ_k^N such that

$$-\frac{\partial J^N}{\partial u^N}(v_{k-1}^N, u_k^N) + \lambda_k^{N\,T}\frac{\partial e^N}{\partial u^N}(v_{k-1}^N, u_k^N) = 0.$$

4. For $n = N-1, N-2, \ldots, 1$, find λ_k^n, such that

$$-\frac{\partial J^n}{\partial u^n}(v_{k-1}^n, u_k^n) + \lambda_k^{n\,T}\frac{\partial e^n}{\partial u^n}(v_{k-1}^n, u_k^n) + \lambda_k^{n+1\,T}\frac{\partial e^{n+1}}{\partial u^n} = 0.$$

5. Finally we find the gradient of $-J$ as

$$-\frac{dJ^n}{dv^n}(v_{k-1}^n) = \frac{\partial L_c}{\partial v^n}(v_{k-1}^n, u_k^n) = -\frac{\partial J^n}{\partial v^n}(v_{k-1}^n, u_k^n) + \lambda_k^{n\,T}\frac{\partial e^n}{\partial v^n}(v_{k-1}^n, u_k^n) = 0,$$

and use this gradient in a gradient-based minimisation procedure to find $v_k = \{v_k^n\}_{n=1}^N$.

Step 2 of the algorithm is in effect the same as solving the forward problem, and when solving the non-linear systems of equations in step 2, we use Newton's method so that we find u_k^n as

- Choose $u_{k,0}^n$. For $l = 1, 2, \ldots$ do until convergence:

  For $n = 1, 2, \ldots, N$ do:

$$u_{k,l}^n = u_{k,l-1}^n - \left(\frac{\partial e^n}{\partial u^n}(v_{k-1}^n, u_{k,l-1}^n\right)^{-1} e^n(v_{k-1}^n, u_{k,l-1}^n, u_{k,l}^{n-1}),$$

  where we solve the linear system of equations with the GMRES method.

In order to find λ^n for the adjoint algorithm we need to solve a linear system of equations in step 3 and in step 4, for which we use the GMRES method. For the outer loop in the adjoint algorithm, we use the LBFGS method.

Work Done in Each Iteration

Let us denote the number of grid cells by M_s, and let $M_w = N_{inj} + N_{prod}$ denote the total number of well segments of injection and production type. By investigating the new KKT optimisation algorithm, we see that we need to solve for each iteration, N linear systems of equations of size $2 \cdot M_s$(step 2 and 3), N non-linear systems of equations of size M_w(step 4) and N linear systems of equations of size $2 \cdot M_s$(step 5). In the adjoint method, we need to solve for each iteration, N linear systems of equations of size $2 \cdot M_s$(step 3 and 4), N non-linear systems of equations of size $2 \cdot M_s$(step 2) plus some calculations needed to find an approximation to the second derivative in the outer LBFGS loop. Thus the amount of work done in each iteration will be greater for the adjoint method than for the new KKT optimisation algorithm, when $2 \cdot M_s > M_w$. For our experiments we have that $(\frac{M_w}{2})^2 = M_s$.

In practice, the computational time required to perform step 4 in the new KKT algorithm is negligible compared to that of step 2, 3 and 5, so that the dominating factor is the time required to solve $2 \cdot N$ linear systems of equations of size $2 \cdot M_s$.

For the adjoint method, the calculation of the derivative and the approximation to the Hessian in step 5 is negligible in comparison with the work done in step 2, 3 and 4. In step 3 and 4 we are solving N linear systems of equations of size $2 \cdot M_s$, and in step 2 the computational time depends upon how many iterations of Newton's method which is required in order to solve the forward problem. Our experience shows that one needs to do between 3 to 5 iterations of Newton's algorithm in order to reach convergence, giving a total of $4N$ to $6N$ linear systems of equations of size $2 \cdot M_s$ to solve in each outer loop of the algorithm. From this, one sees that the new KKT algorithm is from two to three times faster per iteration compared to the adjoint algorithm.

κ_{ro}	κ_{rw}	e_o	e_w	S_{or}	S_{wr}	μ_o	μ_w	ϕ
1.0	0.1	2.0	1.5	0.2	0.2	0.5	0.5	0.2

Table 1: Reservoir simulator constants -equal for all experiments.

	Field 1	Field 2	Field 3
c	$1.25 \cdot 10^7$	$1.0 \cdot 10^7$	$1.0 \cdot 10^7$

Table 2: The value of the constant c in the augmented Lagrangian functional, for the three different permeability fields.

Numerical experiments

We perform herein three numerical experiments. The reservoir dimensions are $450m \times 450m \times 10m$ and we use a 15x15 grid, where the permeabilities in the three different experiments are as in figure 2, figure 3 and figure 4, and other reservoir simulator constants are given in table 1. The three different permeability fields are inspired by those used in Brouwer and Jansen (2004). Along the left hand side of the reservoir there is one injection well, with the possibility to control the injection at each of the individual grid cells along this edge at each time step. Similarly, there is one production well along the right hand side of the reservoir, with the possibility to control the individual production rates at each of the grid cells along this edge of the reservoir at each time step. We inject one pore volume of water into the reservoir, over a period of two years, which is divided into 64 time steps.As initial guess, we use uniform injection and production, such that the individual injection and production rates are equal at all the grid cells which are penetrated by a well. The initial pressure is 190 *bar* and the initial water saturation is 0.2. For the revenue of oil produced, we set $I_o = 80\ \$/m^3$ and set the cost associated with water production to $I_w = -20\ \$/m^3$. Although the revenue constant is very low comparing with the prices of today, it is set deliberately to this value since we wish to make a comparison with the study done by Brouwer and Jansen (2004). The value of the constant c in the augmented Lagrangian functional (12), for the three different experiments, are given in table 2. This constant is found experimentally for each specific experiment, as the one that seems to be best fitted. It is our experience that if one uses a too high value of the constant c, the new KKT algorithm converges very slowly. On the other hand, if one uses a too small value for c, the constraints does not converge to zero such that the algorithm does not converge.

Results

The development of the NPV, for the three experiments with the three different permeability distributions is shown in figures 5, 8 and 11. The NPV is plotted against the number of times the forward problem is solved for the adjoint method, while for the new method it is plotted against the number of outer loops in the algorithm. In all these plots, the NPV for the adjoint algorithm does not change initially. This is due to the line search performed by the LBFGS algorithm which requires both the gradient and the function value of the NPV for each new value of v. In the experiments, the optimisation with the adjoint method stops when the relative change in the NPV is less than a predefined constant while the new KKT method iterates for a fixed number of iterations. One can see from figures 5, 8 and 11 that the two algorithms finds approximately the same net present value in all the three different cases. Moreover, the number of times the forward problem is solved when using the adjoint method is about the same as the number of outer loops made with the new KKT algorithm. However, as discussed previously, one outer loop with the new KKT algorithm is cheaper, in terms of computational time, than solving the forward problem once. The figures 6 and 7, 9 and 10, 12 and 13, show in the top sub-figures the

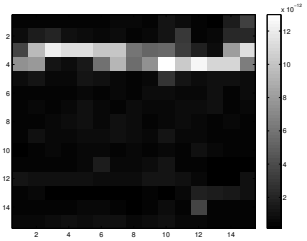

Figure 2: Permeability field 1, given in m^2.

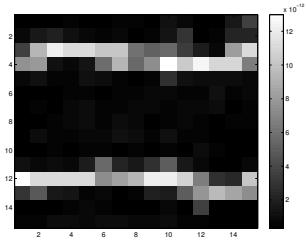

Figure 3: Permeability field 2, given in m^2.

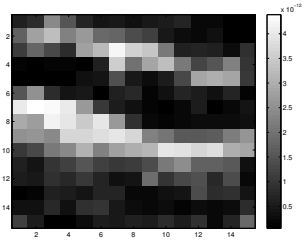

Figure 4: Permeability field 3, given in m^2.

optimum injection rates and in the bottom sub-figures the optimum production rates found with the new method and with the adjoint method, plotted against time step number on the horizontal axis, for the three different cases. We see that the two algorithms do not converge to the same optimum rates, illustrating that there are several different solutions to these problems, and that the two different methods may find different optima.

Conclusions

We have presented a new method for solving a NPV maximisation problem based on solving the Karush Kuhn Tucker equations for the augmented Lagrangian functional. In the examples tested, the new method finds approximately the same NPV value as the adjoint method with less computational effort.

Acknowledgements

This work is financed by the Norwegian Research Council and Total, and we greatly acknowledge their support.

References

Brouwer, D. R. and Jansen, J.-D. [2004]. Dynamic optimization of waterflooding with smart wells using optimal control theory. SPE Journal, 9(4), 391–402.

Byrd, R. H., Lu, P., Nocedal, J. and Zhu, C. [1995]. A limited memory algorithm for bound constrained optimization. SIAM Journal on Scientific and Statistical Computing, 16(5), 1190–1208.

Byrd, R. H., Nocedal, J. and Schnabel, R. B. [1994]. Representations of quasi-newton matrices and their use in limited memory methods. Mathematical Programming 63, 4, 129–156.

Glowinski, R. [1984]. Numerical Methods for Nonlinear Variational Problems. Springer Verlag.

Golub, G. H. and Van Loan, C. F. [1996]. Matrix Computations. The Johns Hopkins University Press.

Lien, M., Brouwer, R., Jansen, J. D. and Mannseth, T. [2006]. Multiscale regularization of flooding optimization for smart field management. Paper 99728-MS, in proceedings of the Intelligent Energy Conference and Exhibition, 11-13 April, Amsterdam, The Netherlands.

Nocedal, J. and Wright, S. J. [1999]. Numerical Optimization. Springer Verlag.

Saad, Y. [2000]. Iterative Methods for Sparse Linear Systems. Yousef Saad.

Sarma, P., Aziz, K. and Durlofsky, L. J. [2005]. Implementation of adjoint solution for optimal control of smart wells. Paper 92864-MS, in proceedings of the SPE Reservoir Simulation Symposium, 31 January-2 Feburary, The Woodlands, Texas.

Zakirov, I. S., Aanonsen, S. I., Zakirov, E. S. and Palatnik, B. M. [1996]. Optimizing reservoir performance by allocation of well rates. 5th European conference on the Mathematics of Oil Recovery, Leoben, Austrian.

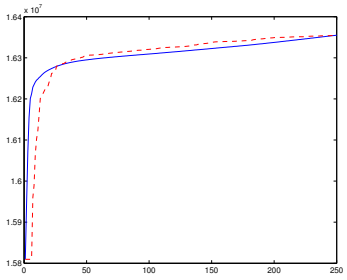

Figure 5: Development of the objective function, for permeability distribution 1. The adjoint method is shown as a dashed-dotted line. The new method is shown as a solid line.

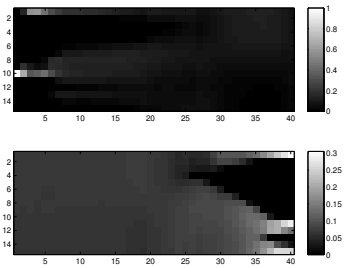

Figure 6: Final rates with the new method, for permeability distribution 1.

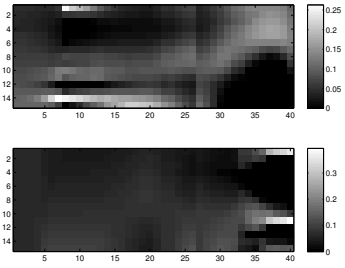

Figure 7: Final rates with the adjoint method, for permeability distribution 1.

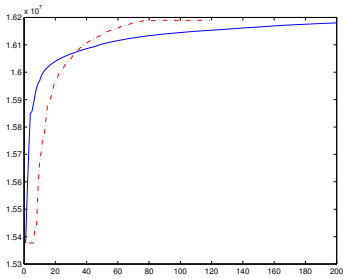

Figure 8: Development of the objective function, for permeability distribution 2. The adjoint method is shown as a dashed-dotted line. The new method is shown as a solid line.

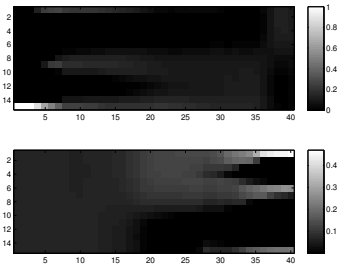

Figure 9: Final rates with the new method, for permeability distribution 2.

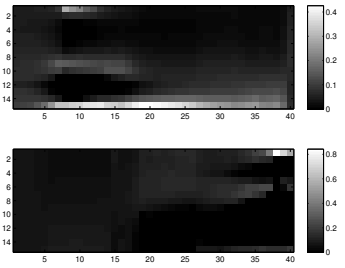

Figure 10: Final rates with the adjoint method, for permeability distribution 2.

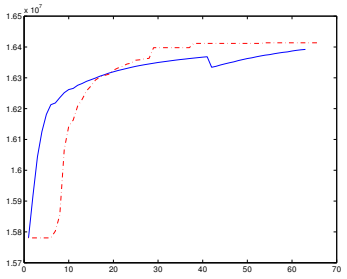

Figure 11: Development of the objective function, for permeability distribution 3. The adjoint method is shown as a dashed-dotted line. The new method is shown as a solid line.

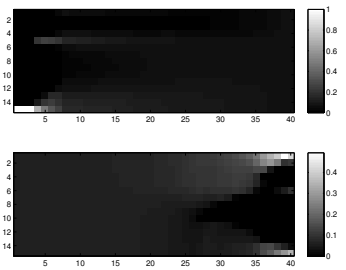

Figure 12: Final rates with the new method, for permeability distribution 3.

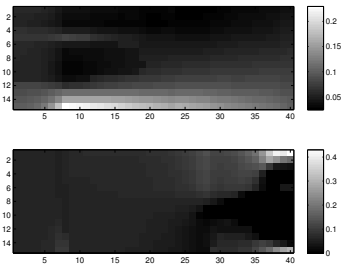

Figure 13: Final rates with the adjoint method, for permeability distribution 3.